



User Guide

for OpenLabyrinth version 3.2.1
Sep 2014

This work is licensed under a Creative Commons
Attribution-Noncommercial-Share Alike 3.0 license

Contents

Contents	2
Glossary.....	6
Welcome.....	7
1. Introduction	7
1.1 What is OpenLabyrinth?	7
1.2 What's new in version 3.1	8
2. Playing a Labyrinth	8
2.1 Open a case	8
2.2 User Interface.....	9
2.3 Bookmarks	10
2.4 Continuing play	10
2.5 Finishing a case	10
3. The Structure of OpenLabyrinth	10
3.1 More on what OpenLabyrinth is.....	10
3.2 Installing OpenLabyrinth.....	11
3.3 Where to find cases to try.....	11
3.4 Permissions.....	12
3.5 Keys	12
3.6 Navigating the Player interface	12
3.7 Labyrinths.....	12
3.8 Nodes.....	13
3.9 Links.....	13
3.10 Rules	14
3.11 Feedback	14
3.12 Elements and Clusters	14
3.13 Avatars.....	14
3.14 Questions	15
3.15 Chats.....	15
3.16 InfoButtons	15
3.17 ImageMaps and hotspots	16
3.18 Presentations	17
4. Creating a Labyrinth	17
4.1 Using the Visual Editor to create a Labyrinth	17
4.2 Using Wizards to create a Labyrinth.....	17
4.3 Creating a Labyrinth by Importing a MedBiquitous Virtual patient Package	23
4.4 Creating a Labyrinth using VUE	23
4.5 Creating a Labyrinth Manually.....	24

4.6 Creating a Labyrinth by Duplicating an existing Labyrinth	24
5. Editing in OpenLabyrinth	24
5.1 Inline Editing.....	24
5.2 Editor Functions	25
5.3 Play	26
5.4 Details	26
5.5 Delete.....	28
5.6 Visual Editor	28
5.7 Nodes.....	30
5.8 Links.....	33
5.9 Finishing a case	33
6. Editing for Advanced Authors	34
6.1 Labyrinth Details.....	34
6.2 User Interface.....	34
6.3 InfoButtons and context-sensitive Help	35
6.4 ImageMaps and hotspots	36
6.5 Node Grid.....	39
6.6 Sections	41
6.7 Chats.....	43
6.8 Questions	44
6.9 Question Rules.....	48
6.10 Avatars.....	49
6.11 Counters.....	50
6.12 Counter Grid.....	51
6.13 Counter displays.....	52
6.14 Rules	56
6.15 Conditional Logic in OpenLabyrinth3.....	57
6.16 Pop-up messages	60
6.17 Elements	61
6.18 Clusters	63
6.19 Feedback	64
6.20 Skin	64
6.21 Files.....	64
6.22 Users.....	66
6.23 Sessions.....	66
6.24 Export.....	66
6.25 Duplicate	66
6.26 Author Notes	67
6.27 Key Feature Problems and Matching	67
7. Feedback and Reporting	68
7.1 Session Reports.....	68

7.2 Feedback Options	71
8. Rapid Reporting of Real-time Responses (4R)	72
8.1 Creating a 4R report	72
8.2 Which choices in a Dandelion?	72
8.3 Creating a Section	73
8.4 Manual edit of a Section	76
8.5 Generating a 4R Report	76
8.6 Looking at more than choice order	78
9. Scenarios	78
9.1 Scenario Manager	78
9.2 Creating a Scenario	78
9.3 Using Scenarios	81
9.4 Scenario-based Learning Design	85
10. Forums	85
11. Global Functions	86
11.1 Users and Groups	86
11.2 Language support	87
11.3 Presentations	88
11.4 Collections	89
11.5 MedBiquitous Virtual Patient Export	89
11.6 MedBiquitous Virtual Patient Import	89
11.7 Migrating cases from OpenLabyrinth version 2 to version 3	90
11.8 Migrating cases between OpenLabyrinth version 3 servers	90
11.9 Importing MVP formatted cases from other software	91
12. OpenLabyrinth Remote Services	91
12.1 Description	91
12.2 Setting up OpenLabyrinth Remote Services	92
12.3 Remote Services Components and Messaging	93
12.4 OpenLabyrinth Remote Services Transactions	94
12.5 Basic OpenLabyrinth Client Functions	96
12.6 OpenLabyrinth Client Enhancements	97
13. Customization: Skins and Mashups	97
13.1 Skins	97
13.2 Working with old format skins	98
13.3 Working with new format Skins	100
13.4 Code	101
13.5 Documentation	101
13.6 Mashups	101
14. Development Techniques	102
14.1 Using VUE	102

14.2 Using Nodes.....	103
14.3 Using Links.....	103
14.4 Using Counters.....	103
14.5 Different Kinds of Labyrinth Designs	103
15. Collaborative Editing as a Team.....	104
15.1 Potential team roles.....	104
15.2 General cautions about collaborative editing in OpenLabyrinth	104
16. Imagemaps and hotspots	105
17. Frequently Asked Questions	108
18. Further Information and Resources.....	109
18.1 References.....	109
18.2 Virtual Patient case libraries	109
19. Appendix 1: Installation	111
19.1 Preparation.....	111
19.2 Code and Directory	111
19.3 Database Setup.....	111
20. Appendix 2: GNU General Public License (GNU-GPL) v. 3.0.....	112
20.1 Preamble	112
20.2 TERMS AND CONDITIONS.....	113
21. Advanced Programming in OpenLabyrinth	121
21.1 Syntax for counter Rules	121
21.2 Event triggers and sequence of Counter evaluations	121
21.3 Basic Syntax.....	121
21.4 Parsing text input.....	122
21.5 Text Validators	122
21.6 Double loop problem	124
21.7 Quirks and Gotchas.....	127
22. Authentication Systems	127
23. Connecting with Other Systems	128
24. Security and OpenLabyrinth	128
25. Table of Figures.....	129

Glossary

Avatar – cartoon representation of a person in the labyrinth

Case – generally synonymous with labyrinth or map.

Chats – simple text expansion displays. When the user clicks on a Chat title, additional text is then shown within the node.

Counter – a simple way of tracking scores or values. In programming terms, this would be considered a 'variable'

Imagemap/hotspot – an image with live clickable areas that then link to other nodes in the case or to an external URL.

InfoButton – a small blue “i” symbol, denoting a link to a popup secondary page of additional information.

Labyrinth – generally synonymous with case or map. A self-contained collection of nodes, along with associated components.

LAMP – Linux Apache MySQL PHP, the combination of components behind the majority of web servers.

LDAP – lightweight directory access protocol, a form of authentication.

MAMP – Mac Apache MySQL PHP, a variant on LAMP

Map – generally synonymous with labyrinth or case.

MVP - Medbiquitous Virtual Patient – an ANSI-standard format, affording case portability between MVP compatible virtual patient platforms.

Node – a single web page within a labyrinth.

Node ID – a unique page number or identifier on that OpenLabyrinth server.

OpenLabyrinth – open source virtual player platform

Presentation – deprecated from v3.11 onwards.

Rules – simple conditional logic that alter the value of Counters or navigation within a case.

SCORM - Sharable Content Object Reference Model. OpenLabyrinth cases have rudimentary SCORM metadata embedded within them.

Skin – a style or theme applied to a labyrinth, allowing the author to change page colors, layouts and fonts across all the nodes in a labyrinth.

Wikiref – a wiki-style reference within node text e.g. [[CR:1234]] refers to Counter 1234. The value of the counter will be substituted into the node page when the labyrinth is played.

Welcome

This guide is intended to act both as user guide and technical documentation for the system. As such, different sections of this guide may be more appropriate starting places for different reader audiences. Please feel free to start at the beginning and read through the guide in a linear manner like a book, or use the following suggestions to help you decide where to begin reading.

If you are completely new to working with virtual patients or OpenLabyrinth, please start at 1. [Introduction](#)

If you are interested in learning how to play through a labyrinth, please start at 2. [Playing a Labyrinth](#)

If you would like to learn more about the features available, please start at 3. The [Structure of OpenLabyrinth](#)

If you are interested in learning how to create your own labyrinth, please start at 4. [Creating a Labyrinth](#)

If you have already created a labyrinth, you can find more advanced options in 6. [Editing for Advanced Authors](#)

If you are interested in 7. [Feedback and Reporting](#) then jump ahead to that section. (Useful for educators.)

Sections 11. Global Functions and beyond are aimed at a more technical audience.

Of course, if you are really stuck, you can always skip to: [Frequently Asked Questions](#) near the end.

1. Introduction

1.1 What is OpenLabyrinth?

OpenLabyrinth is an online activity modelling system that allows users to build interactive ‘game-informed’ educational activities such as virtual patients, simulations, games, mazes and algorithms. It has been likened to an online flexible narrative, similar to the old Choose Your Own Adventure style of book. Depending on what decisions you make or paths you navigate through the case, the consequences will be different. A well-designed case should be challenging and intriguing, with real decisions similar to real-life medicine.

OpenLabyrinth is open source, which means that the original code is made freely available and it may be modified and redistributed (see Appendix 1: Installation for details). It has been designed to be adaptable and simple to use while retaining a wealth of game-like features.

OpenLabyrinth is a web application that will run on most modern web browsers – you do not need any other software on your own machine. You will need access to a server running OpenLabyrinth. See section 18.2 [Virtual Patient case libraries](#) for how to find such case libraries.

For more detailed notes about what an OpenLabyrinth case consists of, see [Editing in OpenLabyrinth](#).

1.2 What's new in version 3.2.1

The last major release of OpenLabyrinth was v2.5 released in November 2009. This new version, v3.2.1, includes many new and upgraded features.

The following are the main changes in version 3.1:

- Scenario based learning design
- Rapid Reporting of Real-time Results (4R)
- Further improvements to Visual Editor
- Further improvements to conditional logic for counters and jumps
- Improved questions, including SCT, SJT format questions and reports
- Support for collaborative team editing
- Authentication links to LTI, OAuth.

2. Playing a Labyrinth

2.1 Open a case

This section describes the steps a user takes to start and work through an OpenLabyrinth case.

Anyone can try out any of the open cases on an OpenLabyrinth site. To start a labyrinth, just click on its title. This loads the first node and displays it (see Figure 1). At the bottom of the page, you will usually see choices on where to go next, like a Choose Your Own Adventure book. Simply click on the link to go to that page.

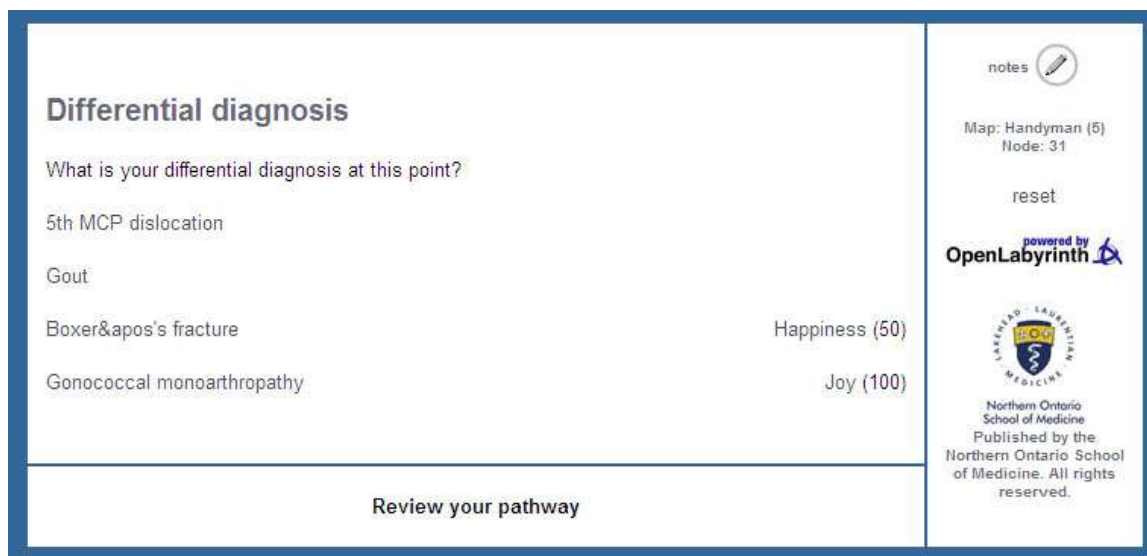


Figure 1: a typical labyrinth screen

To get more out of most OpenLabyrinth sites, you need a login id. This will give you access to more cases, and also to better feedback reports at the end of your case. You can also try out your hand at writing your own simple cases. [Contact us](#) if you would like a trial [login on our demo site](#).

2.2 User Interface

There may be a number of different elements on display (see Figure 2: a typical labyrinth screen's elements).

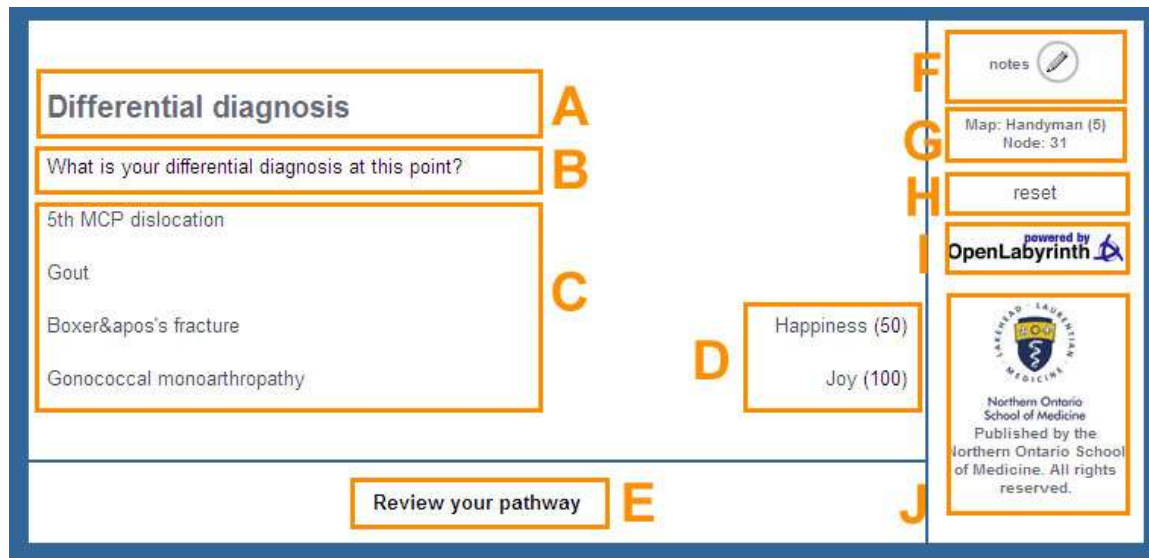


Figure 2: a typical labyrinth screen's elements

A: Node title; B: Node message/content; C: Links/options; D: Counters – label plus current value; E: Link to review nodes viewed in the current session; F: Open note taker; G: Current labyrinth and node information; H: Reset/restart current labyrinth; I: link to OpenLabyrinth home page; J: Skin-specific graphics and text.

A more detailed explanation of each of the different elements is given below:

- Title: every node has a title, which is typically displayed at the top of the page.
- Node content: although not mandatory almost all labyrinth nodes will have some text content for the user. Typically that will describe the consequences of having made the previous choices of paths and possibly things to consider in making the next choice.
- Linked options: the way to navigate an OpenLabyrinth case is to click on one of the available options here. These will usually show the title of the node you are linking to.
- Review your pathway: a clickable track of your pathway through the labyrinth since you started is available. Click on a link to go back to that point – note that this does not roll back the track of which nodes have been visited. Every action is recorded in a session. This may also affect how some Counters and Rules operate.
- Counters: a simple way to show your progress in a game. Authors can show or hide counters on each node, minimizing distracting information when not needed. There may also be a display of how the counter's value has changed since the last node.
- Map and Node ID: this shows the current labyrinth and node ID number. The node ID is a unique identifier that can be thought of like a page number for that node. If there is a problem with a case, the case authors will appreciate you making a note of which node ID number caused your problems.
- Reset link: this ends your present session and restarts you in a new session within the same OpenLabyrinth case.
- A link to OpenLabyrinth home page: clicking the OpenLabyrinth icon returns you to the home page.
- Other graphics, links, tools and text may also be displayed depending on the current skin.

2.3 Bookmarks

Sometimes a labyrinth may be too long to complete in one session or perhaps a learner only has a little time. The bookmark feature (if enabled in the Skin) allows you to set a bookmark at a certain point within a labyrinth. To restart from a stored bookmark go to “My Labyrinths” menu from the home page. If you see an orange [Resume] button for a labyrinth, click the button to carry on from that point. Note that only the last bookmark in any given session is stored. You can also Play a labyrinth, with a stored bookmark, without upsetting or resetting its current location. Bookmarks are personal to each individual logged-in user.

2.4 Continuing play ...

Generally, navigating your way through a well-designed OpenLabyrinth case should be reasonably intuitive. Many cases follow a fairly linear format, often in the HEIDR (History, Exam, Investigation, Diagnosis, Rx or management) model. But OpenLabyrinth authors are not at all restricted to this style. OpenLabyrinth provides an environment in which cases can have many different styles – multiple endings, varying levels of complexity, and can support some simple games where the flow is determined by the game designer. See Section 14.5 for some more information about case designs and topology.

What you experience depends on the design of the labyrinth you are playing and the choices available within it but typically you make decisions as to which path you will take and these decisions have different consequences depending on which nodes are passed through. Scores and counters may go up or down, paths may be dead ends or choices may end the current activity, while other paths will be successful. When you are playing a labyrinth you should think carefully about the options available to you, keep an eye on any counters or timers in the activity and follow the instructions and hints given you.

2.5 Finishing a case

Most cases will make it clear when you have finished (saved your patient, killed them etc.). You might be invited to reflect on your scores in the Counters. Some cases will provide the option to have a more detailed report on your performance. You must be logged in to the OpenLabyrinth server at the start of the case for this.

Every option selection (or ‘click’ - along with the current score, timer and counter values) is recorded as you work through a labyrinth. This tracking supports the pathway review function as well as being able to generate a report on how you did within the activity.

If the current labyrinth has been set to provide a feedback report then at some point you will be presented a link that says “**end session and view report**” – this will end the current session and provide a comprehensive feedback report – see [Feedback and Reporting](#) for more information.

3. The Structure of OpenLabyrinth

3.1 More on what OpenLabyrinth is

As a player of cases, you don’t really need to understand what is going on behind the scenes. Often, it is better to hide the complexity of a case behind a nice simple interface. You simply need to work your way through a case, clicking on links, answering questions and quizzes. But if you plan to write your own cases, it does help to understand the basic anatomy of an OpenLabyrinth virtual patient case.

OpenLabyrinth is an open source online activity modelling system that allows users to build interactive ‘game-informed’ educational activities such as virtual patients, simulations, games, mazes and algorithms. It has been designed to be adaptable and simple to use while retaining a wealth of game-like features. OpenLabyrinth is licensed under the GNU General Public License v3 (GNU-GPL3: www.gnu.org/licenses/) - see [Appendix 2 in section 19](#)

The original Labyrinth application was originally developed by the Learning Technology Section of the College of Medicine and Veterinary Medicine at the University of Edinburgh, which was subsequently revised into OpenLabyrinth by Rachel Ellaway at the Northern Ontario School of Medicine along with input from partners in Canada, the UK, Germany, Australia and the US.

OpenLabyrinth v3 was completely rewritten under the guidance of the OpenLabyrinth3 Consortium, lead by the University of Calgary, as an open-source project. The code was completely rewritten in PHP, obviating the previous problems arising from dependencies on closed 3rd party code libraries. The main version now is based on Linux, Apache, MySQL and PHP, rather than using Microsoft Windows servers and software. Because the source code is openly available, it can be recompiled for similar platforms, including Windows servers. Further information about this is available in section 19 [Appendix 1: Installation](#).

We have found OpenLabyrinth’s openness and flexibility is very useful for creating a variety of different activity designs, going way beyond isolated virtual patient cases. Some examples are:

- Group work cases
- Hybrid simulations with Standardized Patients
- Bookending high fidelity simulations
- Video Mashups
- Resource trade-off scenarios
- Multi-case triage
- Mini-cases as mobile exemplars
- Script Concordance Testing
- Bidirectional integration with podcasts and webinars

3.2 Installing OpenLabyrinth

OpenLabyrinth is a web application that will run on most modern web browsers – you do not need any other software on your own machine to play a case. In order to create and edit your own cases, you will need access to a server running OpenLabyrinth. This runs on Apache and MySQL, the most commonly used platform for web-based applications. For demonstration and test purposes, you can request access to our demo server at <http://demo.openlabyrinth.ca> by emailing us at info@openlabyrinth.ca - this server runs the most recent version of the OpenLabyrinth code. It is not recommended for production level cases. For this we recommend that your organization set up its own server. See [Appendix 1: Installation](#) for more details on how to install and configure OpenLabyrinth.

3.3 Where to find cases to try

There are many libraries of OpenLabyrinth virtual patient cases available where you can try running a case to see if it might fit your needs. Firstly, try some of the open cases available at our main server, <http://vp.openlabyrinth.ca/> or see section 18.2 [Virtual Patient case libraries](#). A more updated list is available at our web site: <http://openlabyrinth.ca> - please let us know if you come across other accessible cases that are not on our list. Many of these libraries will feature older version 2 OpenLabyrinth cases but these will still give you an idea of what can be done with this application. OpenLabyrinth v3 retains full compatibility with the ANSI/[Medbig Virtual Patient \(MVP\) standard](#), which means that it will run OpenLabyrinth v2 cases, and also virtual patients from other virtual patient engines that are compliant with the MVP standard.

Some cases that we recommend you look at first include:

- [VP on VPs](#) – a virtual patient on virtual patients
- [Welcome to OpenLabyrinth](#)
- [Harriet Headcase](#) – a case designed in 3 parts for group work
- [IUCD Mashup](#) – illustrating how OpenLabyrinth can be used to splice video segments
- [Chester Angermeier](#) – an example of a case ported from the eViP repository
- [Mildred Blonde](#) – example of a vague and dizzy historian
- [John's back again](#) – written by one of our residents
- [Teaching Tips](#) – a placeholder for linked information. Not a patient.
- [Counting on You!](#) – showing how counters can be used. For authors.
- [Medical Careers](#) – based on a classic board game

3.4 Permissions

A labyrinth's security settings can be set at four different levels:

'Open' labyrinths can be played by anyone on the web, and logging in is not necessary to access them.

'Closed' labyrinths can be played by anyone who is logged into that OpenLabyrinth server.

'Private' labyrinths can only be played by their authors, or users given access by those authors

'Key' labyrinths can only be played if you know a preset keyword, assigned by the author.

See also [Users and Groups](#) for more information about how this works along with user access levels.

3.5 Keys

You can also require your users to enter an arbitrary text key to activate a Labyrinth. To turn on the use of keys set the labyrinth security type to "keys" in the Details editor and use the 'edit' link by the security type select to create one or more keys. A key can be any kind of text string including variations such as 'green for go' and '85A94W8BA9445'. When a user tries to run a labyrinth they will be challenged to enter a valid key and they won't be able to start until they do so.

Keys are particularly useful when you want to quickly turn on or off when a case can be accessed, for example for workshops or exams.

3.6 Navigating the Player interface

This has already been described – see section 2 [Playing a Labyrinth](#).

3.7 Labyrinths

A 'labyrinth' is the principal unit of organisation within OpenLabyrinth. Each labyrinth has a series of global properties such as the type (game, maze, algorithm etc.), its authors, timers, visual appearance (skins), security, scores and counters etc. Within each labyrinth there are a series of linked pages or 'nodes' that define the options available to you, each of which can be enhanced with a number of behaviours and services to further structure your experience and gameplay. Confusingly, you will also see the terms 'map' and 'case' used to refer to a labyrinth, within the OpenLabyrinth community.

3.8 Nodes

A labyrinth is made up of many interconnected nodes. A node is one unit of presentation to the user; each node is displayed as a single web page. Figure 3 shows a diagrammatic representation of a simple activity consisting of six nodes. A node is the main unit of a labyrinth, and all the other labyrinth components are organised around the nodes.

Every labyrinth node has a unique identifier (node ID) as well as a number of other properties such as a title, node content (text-based), a type (root or child) and a series of rule and function properties. The root node is the labyrinth starting point and there is only one node of this type in each labyrinth. The rules and functions determine what content is presented to you at any given node and this will depend on both current node properties and what you have done previously (particularly concerning scores and counters). Other node properties include whether the node must be visited or avoided and whether you can end the session and see a report of how they did.

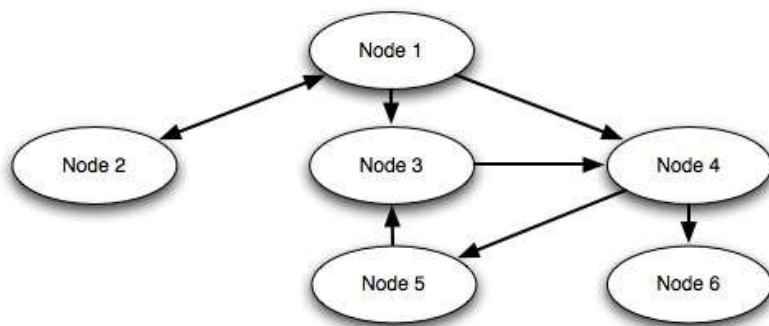


Figure 3: nodes and links

Node 1 is linked to nodes 2, 3 and 4, node 2 is linked back to node 1, node 3 is linked to node 4, node 4 is linked to nodes 5 and 6, node 5 is linked to node 3 and node 6 isn't linked to anything.

3.9 Links

Labyrinth nodes are connected by a series of links expressed as pairs of node IDs with some additional properties such as ordering, icons and alternative text. Because links are one directional (from node A to node B) a link back requires a second link (from node B to node A) with independent properties. There are 8 links indicated in figure 3 because there is a link from node 1 to 2 and another separate one from node 2 to 1.

Links can be presented to you in five different ways

- Hypertext – clickable text link per available choice
- Drop down – a dropdown list of the available choices
- Dropdown with confidence interval – as above with a second drop down indicating how confident (or unconfident) you are about your choice
- Type in text – you type in the first few letters of your proposed answer and if it matches an available option this is auto filled for you.
- Buttons – similar to hypertext links, but displayed as buttons, with the destination node as title.

The order of these available options can be set, randomized or randomly set to present just the one option.

3.10 Rules

Rules are functions attached to nodes that change the way a labyrinth is presented to you. One set of rules might set or change one or more counters while another could require you to have visited other nodes before being able to load the current one. In a little more detail these rules include:

- Counters: there can be as many or as few counters in a Labyrinth as you want. Each counter has a name, description, starting value and a number of functions. At the entry to any node or on clicking any link the value for each and every counter can be changed using plus, minus or equals operators along with an integer value; '+10' adds ten, '=4' sets the value to 4 irrespective of its previous value. These values can then trigger rules based on the current value of each counter.
- Conditionals: these control access to a node based on which nodes you have visited previously. For instance a rule that looks like "{15}OR{16}" would mean that you couldn't enter node 17 without having visited nodes 15 or 16 first. Each rule is made up of node IDs connected with standard Boolean operators.
- Timers: these provide a real time countdown at the end of which you need to restart the activity. A 300 second timer would mean that you would need to complete the activity in 5 minutes. Timer functions are not yet fully implemented in OpenLabyrinth v3.

In OpenLabyrinth v3, we have instituted a more powerful approach that allows the case author to apply some simple IF.. THEN.. ELSE logical structures to how the case is portrayed. This is potentially very powerful and has already opened up some interesting development possibilities for us. The syntax is inherently simple but see section [Advanced Programming in OpenLabyrinth](#) for a more detailed description and how to use them.

3.11 Feedback

The user can be given extensive feedback based on their choices in playing a labyrinth. This includes a report of which choices were made, the counter values and whether nodes were marked as 'must visit' or 'must avoid'.

There are a number of author-configured feedback rules including:

- Feedback per node visited
- Feedback depending on the numbers of 'must visit' and 'must avoid' nodes visited
- Feedback on the time taken to complete
- Feedback on values of counters at the end of the session

3.12 Elements and Clusters

OpenLabyrinth has been designed to import to and export from the MedBiquitous virtual patient data standard, an emerging specification for the exchange and reuse of virtual patient activities between different authoring and player systems. For more information, see the section on [Elements](#).

3.13 Avatars

One way of enhancing a narrative is to identify human characters or agents within its flow. OpenLabyrinth supports the use of characters by providing support for simple animated avatars for these characters in the narrative. An avatar can be configured to appear differently (age, skin, hair, clothing, context) as well as communicate using speech or thought bubbles. The same avatar can be reused within different settings or many different characters can be used.


3.14 Questions

OpenLabyrinth users can control how an activity plays out by selecting from the available choices along the way. Sometimes however some interaction is needed that doesn't advance the story. Questions can be embedded within a node. These can be of several types including open text entry, multiple choice, sliders etc. See here for more advanced info about [Questions](#). Some of these Question types have the ability to modify a score by controlling a counter.

3.15 Chats


These are similar to Questions in appearance. They are simple way of providing extra information to you, if they click on them. See here for more info on [Chats](#)


3.16 InfoButtons

These are also a simple way of providing additional information on a node or page. They appear to you as a simple blue "i" logo. 

Multiple InfoButtons

The primary purpose of this is to show that you can have more than InfoButton on a page and that you can call InfoButtons from other nodes, not just the one belonging to this node.

2nd node Info: 

3rd node info: 

Again, more text can appear below.

Standard InfoButton calling
Root Node

Figure 4: a node can show more than one InfoButton

When you click on the icon, a small sub-window appears with additional information. InfoButtons are generally connected to their parent Node but you can call an InfoButton from another Node.


Edit "Multiple InfoButtons" in Labyrinth "Help and InfoButtons"

Set as Root

Node Content

Title:

Node Content



The primary purpose of this is to show that you can have more than InfoButton on a page and that you can call InfoButtons from other just the one belonging to this node.


2nd node Info: `[[INFO:2339]]`

3rd node info: `[[INFO:2340]]`

Again, more text can appear below.

Path: p

Supporting Information



More infoButton text from the 3rd node

Path: p

Supporting Information
Keyword:

Figure 5: edit InfoButtons using the Supporting Information field

Note that you can place the InfoButton anywhere within the Node Content. Use the `[[INFO:xxx]]` style of reference, just as you would for images and Questions. Unlike OpenLabyrinth v2, where you only had to place some text in the Supporting Information field, you still have to place this marker or else the InfoButton will not show up.

3.17 ImageMaps and hotspots

Within the Node Editor, OpenLabyrinth now gives you the ability to create hotspots on images that act the same as links. These can be links to other nodes, just as with any other OpenLabyrinth link, or can be hyperlinks to any web address. This can be used to extend the visual metaphor of the OpenLabyrinth interface, providing things like virtual EMR screens, clickable hotspots on x-rays or other diagnostic images, or any other clinical image where you want the learner to select a choice. See the section [ImageMaps and hotspots](#) for more details on how to set up an imagemap.

3.18 Presentations

In OpenLabyrinth v2, a presentation was an arbitrary set of labyrinths grouped together for users to access together. This may be a set of materials for a particular course, an examination paper or a group topic. These have been deprecated in OpenLabyrinth v3.2 onwards. We are now using [Scenarios](#) and [Collections](#) to provide similar or greater functionality.

4. Creating a Labyrinth

There are several ways to create a labyrinth within OpenLabyrinth. We will expand on each of these five methods in the sections that follow.

- Using the Visual Editor to create a Labyrinth
- Using Wizards to create a Labyrinth
- Creating a Labyrinth by Importing a MedBiquitous Virtual patient Package
- Creating a Labyrinth Manually

You must be logged in as an author on your OpenLabyrinth web site. If you do not have a local OpenLabyrinth server running, [contact us regarding a guest login](#) on one of our systems.

4.1 Using the Visual Editor to create a Labyrinth

The Visual Editor is a simple concept-mapping tool. It is similar to the VUE concept-mapping tool from Tufts, which we previously used for case design. The OpenLabyrinth Visual Editor is now completely integrated into the rest of the OpenLabyrinth authoring environment. See this section for more on how to do this – [Using the Visual Editor](#)

4.2 Using Wizards to create a Labyrinth

There is now a set of simple wizards that can help you with the initial steps in designing a case. More experienced authors might find it quicker to use more direct methods but this does help to get new authors with getting set up with basic structures of a case.

The screenshot shows the 'Step 1. Add global information' form in the OpenLabyrinth application. The form includes the following fields and options:

- Title:** A text input field containing 'Six steps in case creation'.
- Description:** A text area containing 'Using the wizard to create a linear case'.
- Keywords:** A text input field containing 'test'.
- Timing:** Radio buttons for 'On' and 'Off'. The 'On' option is selected. A 'Timing Delta' field is set to '0' seconds.
- Security:** Radio buttons for 'open access', 'closed (only logged in Labyrinth users can see it)', 'private (only registered authors and users can see it)', and 'keys (a key is required to access this Labyrinth)'. The 'closed' option is selected.
- Section Browsing:** Radio buttons for 'don't show', 'visible', and 'navigable'. The 'visible' option is selected.

At the bottom right, there are two buttons: 'Save & return later' and 'Step 2 - Kind of VP'.

Figure 6: first step in wizard to Create a case

To get started with case design using a wizard, select 'Create Step-by-Step' from the Labyrinths top menu. You will first be asked for global information for the case. All of the information you enter here and subsequently can be edited later so don't worry too much about getting things right just now. Add a title, short description of your case, and appropriate keywords if you wish. You can ignore Timing choices for now. Click on 'private' for Security. Move on to Step 2.

In Step 2, choose what pattern or learning design you want to use. The commonest to start with is a linear or HEIDR case but the power of OpenLabyrinth lies in branched cases. [See section 14.5](#) for examples of some common pattern designs. For this example, we will choose a Linear pattern.

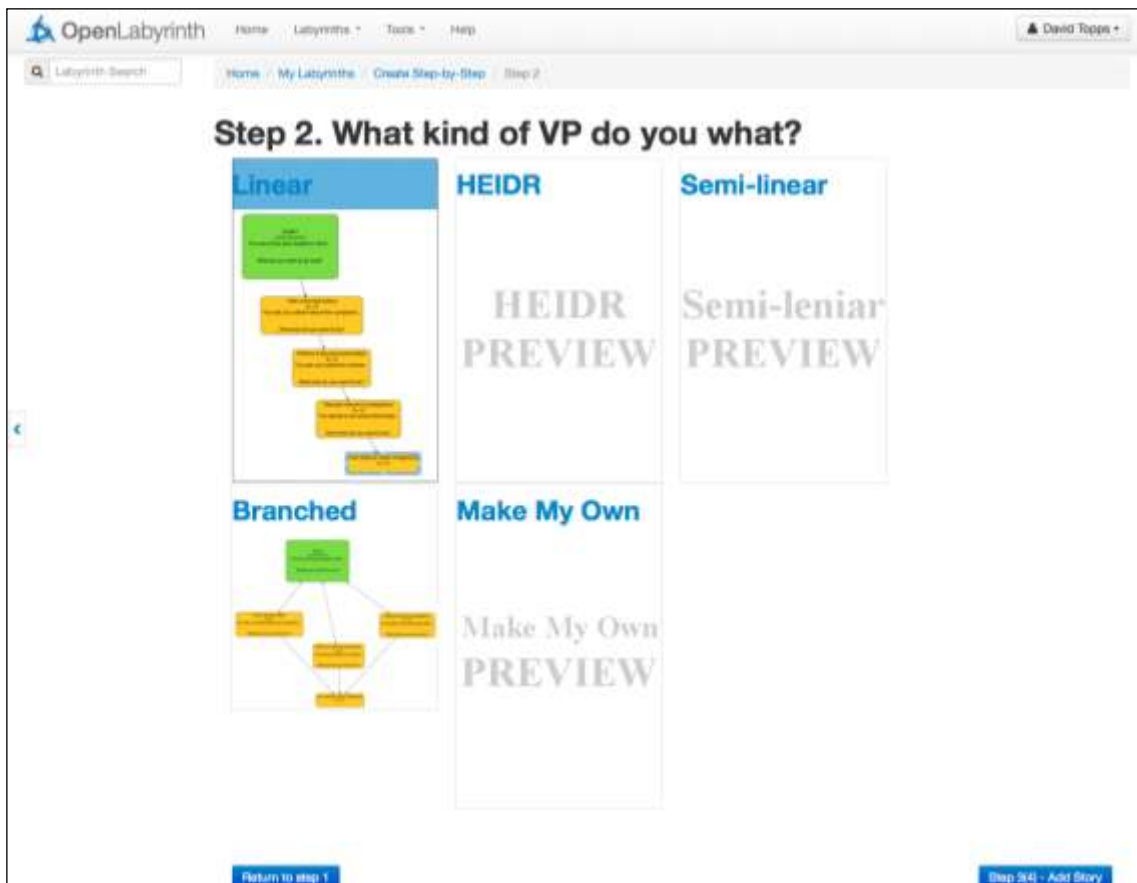


Figure 7: select the kind of pattern you want to use

Go to Step 3 to add your story parts. In this step, we will flesh out the key points to get across. We have chosen a HEIDR style of case, typical for an undergraduate medical presentation: History, Exam, Investigations, Diagnosis and Treatment (Rx). It is good at this point to have a clear idea of the 3 to 4 main teaching points that you want to illustrate. Enter these in the nodes you see here. You can always come back later to flesh out the story more, provide a greater variety of choices, more context and background etc, but getting these main points down now in the backbone of your story will help to keep things focused.

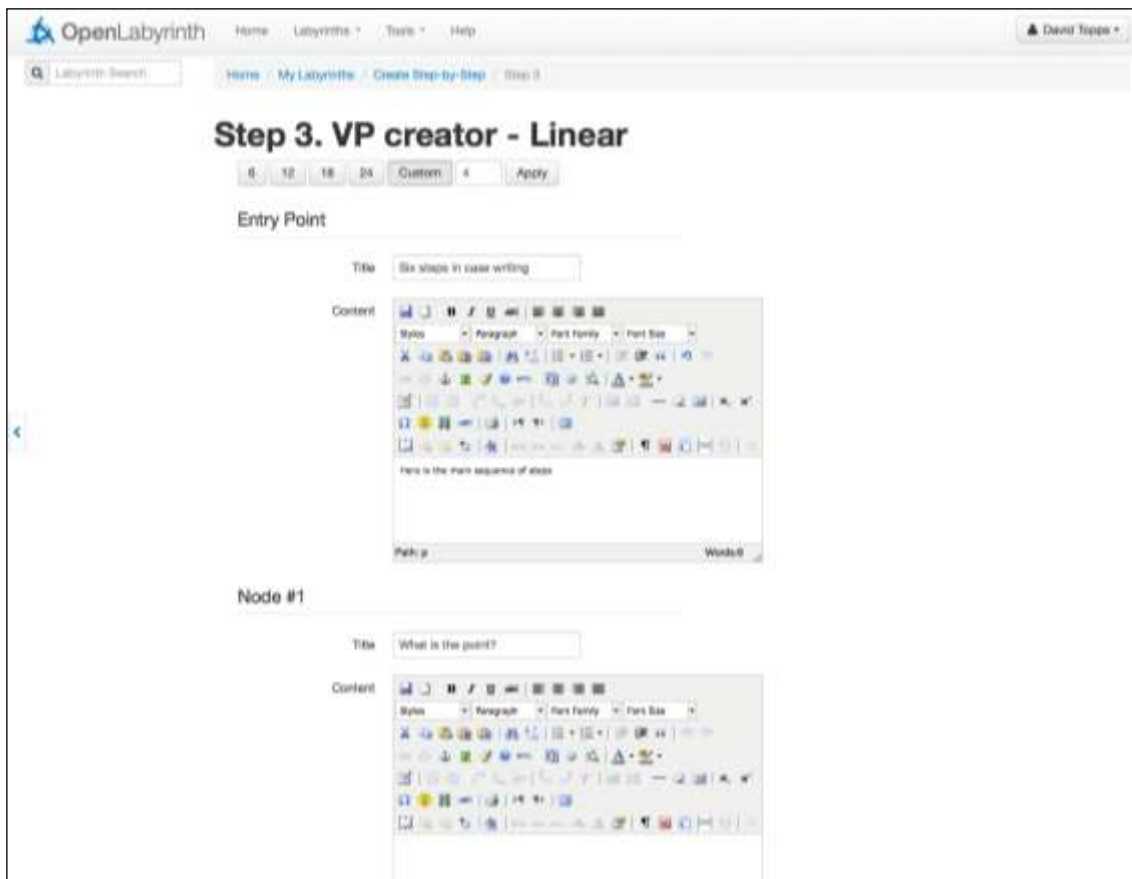


Figure 8: Select number of nodes then fill in bare essentials

Fill out the nodes in Step 3 as an overview then proceed to Step 4 to edit your story points further.

Click on the blue button at bottom right to proceed to Step 4. This will bring up the Visual Editor, showing you your linear pattern, with the information you just inserted.

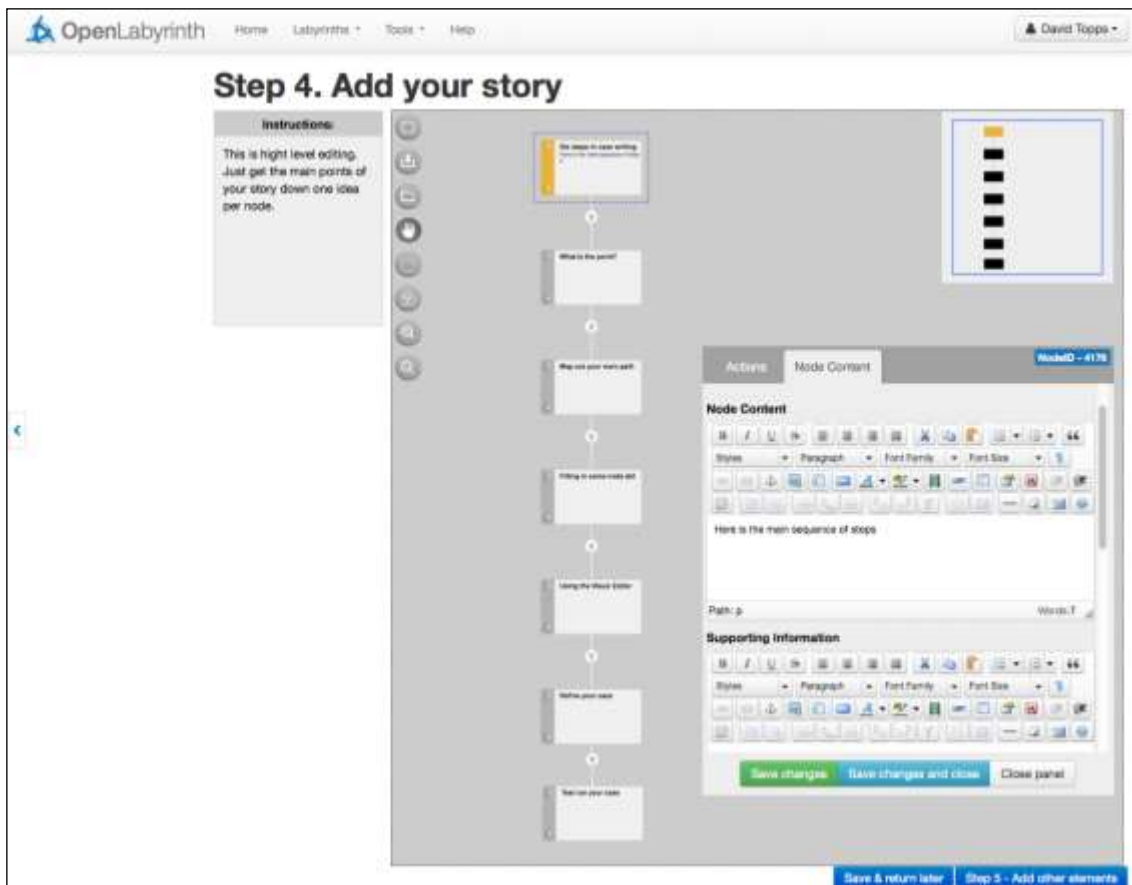


Figure 9: using the Visual Editor to continue mapping the pattern

Add a few additional side nodes for your case here. Don't feel you have to do it all now. You can come back and add more later.

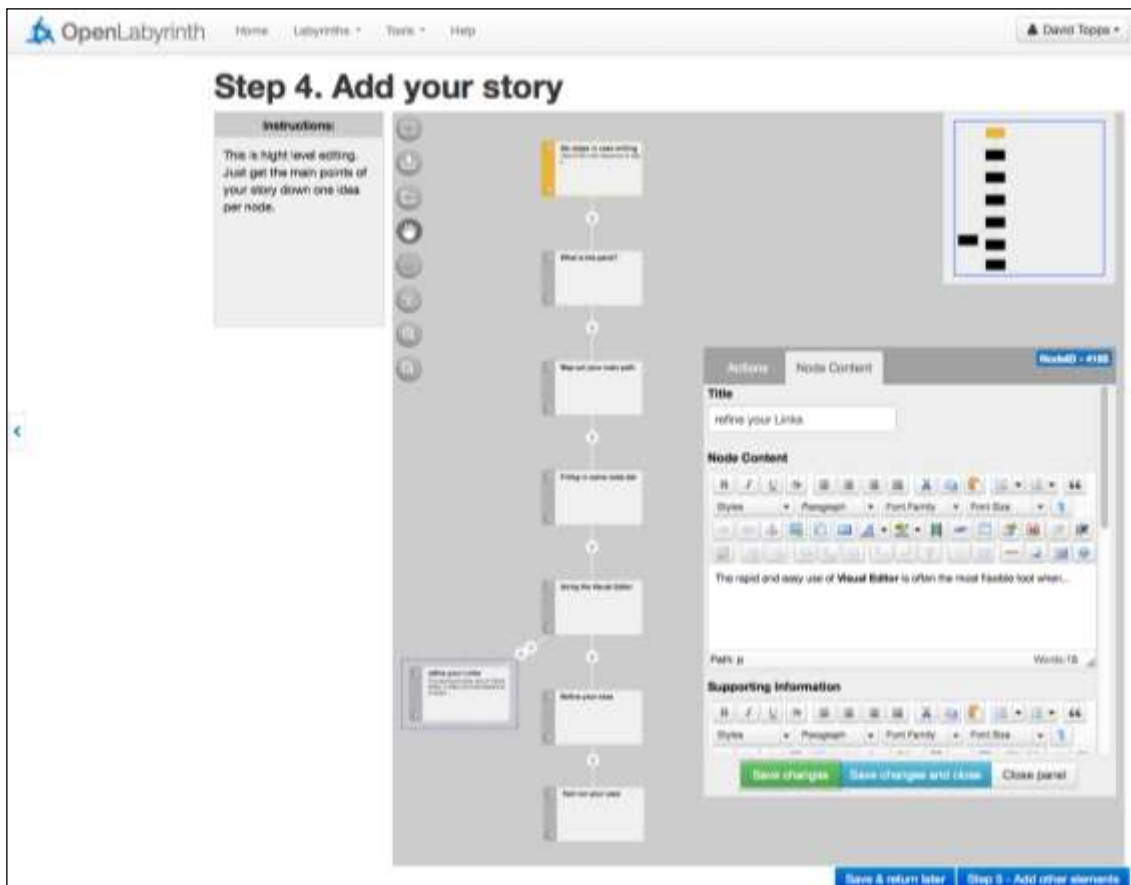


Figure 10: adding a small side branch using Visual Editor

In Step 5, you can add other parts to your case such as pictures, avatars, or insert some brief multiple-choice type questions.

Step 4. Add elements

[Edit nodes](#)
[Add image or file](#)
[Add question](#)
[Add avatar](#)
[Add counter](#)

Nodes: Choose some node in left column.

Your situation
[root] (3304)

Ask pertinent questions (3305)

Perform focused physical exam (3306)

Investigations - pick some (3307)

Differential diagnosis (3308)

Rx options (3309)

Follow up (3310)

Summary (3311)

Cardiac exam (3313)

Abdominal exam (3315)

Neuro exam (3317)

[Return to step 4.](#)
[Save & return later](#)
[Step 5 - Edit Skin](#)

Figure 11: add remaining components [here](#)

In the final step, you can create or select a Skin that fits the theme of your case. You can also do this at any time in the future. For now, you might want to stick with the basic skin and focus on your case content.

Click on 'Save & return later'. You can now go back to edit your case further using a variety of editing tools. See section 5 [Editing in OpenLabyrinth](#) for more details on how to do this.

4.3 Creating a Labyrinth by Importing a MedBiquitous Virtual patient Package

OpenLabyrinth can create new labyrinths by importing MedBiquitous Virtual Patient packages. The package and the import process are described more fully in [MedBiquitous Virtual Patient Import](#).

4.4 Creating a Labyrinth using VUE

VUE is a Java-based visual concept-mapping tool from Tufts University that can be downloaded (for free) from <http://vue.tufts.edu> for both Windows and Mac. You can use it to create designs for labyrinths by creating boxes to represent nodes and the links between them. Although VUE supports many other features, only the boxes (converted to nodes), text in the boxes/nodes and the links (between nodes) will be imported. Everything else will be ignored when the VUE map is imported into OpenLabyrinth.

This feature is **not** currently directly supported in OpenLabyrinth v3. You can still use VUE but the process is now more convoluted. You import your VUE map into OpenLabyrinth v2, export as an MVP format file and then import the MVP file into OpenLabyrinth. Please consult the [OpenLabyrinth v2 manual](#) on how to best use VUE to construct a map that can be smoothly imported.

4.5 Creating a Labyrinth Manually

This is where each node is entered manually, one at a time, along with the links, rules etc. The steps for authoring and editing are very similar other than for authoring you need to create a labyrinth first. To do this first click the 'add a new Labyrinth' link (available to logged in users only) and then fill in the form to set the new labyrinth's global properties.

Once you have created the new labyrinth you can then start building it up by adding nodes, links between nodes and counters, files, rules and other dynamic properties. A single root node will have been created as part of the labyrinth but all other nodes, links, rules etc. will need to be added manually – see section on [Editing in OpenLabyrinth](#) for more details on how to do this

4.6 Creating a Labyrinth by Duplicating an existing Labyrinth

Any labyrinth you have edit access to can be duplicated as a template for a new labyrinth by clicking its 'duplicate' link on the editor page. A basic copy of the original labyrinth is created (called "Copy of ...") allowing you to change any aspect while keeping the original untouched. Note that duplicating just takes the basic structure and does not import files, rules or other additional properties.

At present, this is not working fully. We suggest that you use the slightly more laborious but more complete process of Exporting your labyrinth as an MVP zip file and then Importing that again. OpenLabyrinth will rename the file when importing and will make copies of all the associated resources and media files. [See 6.18 and 6.22 below](#)

5. Editing in OpenLabyrinth

All aspects of a labyrinth can be edited by anyone who is logged in with edit permissions for that particular labyrinth. Simple content editing can be carried out within a node as the labyrinth is being played using the inline editor, while more advanced features are available through the other labyrinth editors.

5.1 Inline Editing

The content of a node can be edited from within that node while a case is being played, if the current user has authoring rights to the current labyrinth. Click the link saying 'turn editing on' to switch into inline editor mode. You will be able to change the current node's title and content from here or click directly to the more advanced editing tools (see figure 12).



Figure 12: inline node editing:

A is the link to turn editing on and off, B is the field to change the title, C is the field to change the text, and D is a series of links to the main editor tools

5.2 Editor Functions

When you open your case, the first page that opens is the 'Details' page

Labyrinth Info	
title	Chester Angermeier for workshops
authors	Sonya Lee (slee), Alaa Aboulhoda (aabolhoda), Peter Colbert (pcolbert), Nishan Sharma (nsharma), Demo One (demo1), Demo Five (demo5), Demo Six (demo6), Demo Seven (demo7), Demo Eight (demo8), Sarah Topps (stopps), Sarah Topps (stopps)
keywords	
Labyrinth type	Maze - no scores, 1 or more startpoints, no endpoints
security	open access
number of nodes	103
number of links	419

Labyrinth Title	<input type="text" value="Chester Angermeier for workshops"/>
Labyrinth Description	<input type="text" value="Based on Mr Angermeier from SGUL. For ICRE workshop Tweaked for Canada. Crude text match on prov Dx."/>
Labyrinth Keywords	<input type="text"/>
Labyrinth Type	<input type="text" value="Maze - no scores, 1 or more startpoints, no endpoints"/>

Figure 13: the OpenLabyrinth properties editor, with common actions in sidebar on Left side.

The 'Details' page is one of 23 main editing functions, which are accessible via the left-hand panel. A brief description of each editing function is given below, in the same order as they appear in the side panel:

- **Play** – this launches the map to run in a new window
- **Details** - this is a duplicate of the form used to create a map manually
- **Delete** – deletes the current labyrinth
- **Visual Editor** – visual node and link editor, in concept-mapping style
- **Nodes** – you can add nodes or edit any existing nodes from here
- **Node Grid** – you can edit all of the node titles and contents within a single Labyrinth at once
- **Links** – you can add links or edit links from here
- **Sections** –create arbitrary sections for grouping nodes together and assign nodes to these sections.

- Chats – text items that expand within the node page to provide further data.
- Questions – embed questions for tracking and evaluation
- Avatars - add and edit different avatars for the current labyrinth
- Counters – you can add and edit counters from here
- Counter Grid – you can see and edit all counter functions for all nodes in a Labyrinth at once
- Counter displays – you can embed the counter values within an image, like a skin
- Rules – you can use simple IF..THEN..ELSE type logic to alter how the case plays.
- Pop-up messages – you can have small popup windows, timed to display after certain events
- Elements – you can add and edit data elements from here
- [Clusters](#) – you can add and edit data clusters from here
- Feedback – you can create and edit feedback rules from here
- Skin – you can alter how the case appears, with differing colors, fonts, backgrounds and layouts.
- Files – you can upload and manage any files you need in your map from here
- Users – you can add or remove users (both authors and learners) for the current Labyrinth from here
- Sessions – reports on all user sessions for the current Labyrinth - see section [Session reports](#)

Initially case authors will only need to use **the most basic editing functions**. These basic editing functions are **bolded in the list above**, and are described in sections 5.3 – 5.8 below. The more advanced editing functions such as the Node Grid, and the sub-options, case design options and controls are described in Section 6: Editing for Advanced Authors.

5.3 Play

This starts the labyrinth running in a new window. You can also start Play from a particular node by clicking on its name in the Nodes editor listing. If the map has not already been played in this authoring session, you may get an error from the Kohana database.

5.4 Details

This page contains the master set of metadata about this labyrinth. What you will see depends on whether you are in Easy or Advanced UI mode. We will discuss the Advanced version here. If your view does not correspond, then toggle your UI using the menu item 'Help | Advanced UI'.

This is the main page where you can edit the properties that structure the whole labyrinth including:

- Title
- Description – this is a rich text field, soon to be included in semantic indexing. Try to be concise as this field will also be used in future case repository descriptor fields.
- Keywords – no structure is currently enforced for these. Use these as you see fit.
- Skin – the current Skin being applied to this labyrinth.

In the Users sub-section of the Details page, we have:

- Security – drop-down list of open, closed, private and keys. See below or [Permissions](#)
- Contributors – while similar to Users/authors, this area allows you to provide greater detail about the level of contribution made. **Note:** if you have several contributors to add, click the blue [+Add] button several times, once for each contributor, before you start editing the contents of this sub-section.
- Creator – the author who first created the case. (This is relevant for certain SQL database functions).
- Registered authors and groups – this is a brief listing of who has been added as authenticated Users for this labyrinth. To edit this, click on the 'Users' link on the left-side menu.

In the Navigation sub-section of the Details page, we have:

- Link function style – a drop-down list where you can choose how users generally move from page to page in the labyrinth. This can be modified on a page by page basis, using the Links editor, but note that this global setting may over-ride such customization. See [Links](#)
- Section browsing – this determines whether the Section headings show up on the page (if allowed by the current Skin layout), and whether they can be used to move around within the labyrinth.
- Timing – this relates back to older functionality within OpenLabyrinth and has since been overtaken by Timed Popups. This function is little used now.

If there is a discussion Forum associated with the labyrinth, it is indicated here. You can also assign a new Forum to the case from here.

In the Reviewer Information sub-section of the Details page, we have metadata that helps authors and reviewers to prepare their case for formal submission to a learning object repository, such as MedEdPortal:

- Link logic verified – toggle plus date. Does the navigation and pathway continuity work as intended?
- Node content verified – toggle plus date.
- Clinical accuracy verified – toggle plus date.
- Media content complete – toggle plus date.
- Media copyright verified – toggle plus date.
- Semantic metadata – toggle plus date.
- Instructor guide complete – toggle plus filename of document in the labyrinth's Files section
- Author notes – additional annotations for the collaborating team, not for consumption by the learning object repository.

In the Metadata sub-section of the Details page, we have an area that can be modified per server, for use along with any available semantic indexing tools. This is in development, in conjunction with Aristotle University, Thessaloniki.

Remember to save your changes using the [Save changes] button at the top or bottom of this page.

Here is a bit more information about how the security levels work for an individual labyrinth:

- Security: can be set – see table 1: security settings and their meaning

Security type	View/run	Edit	Duplicate
Open	Yes	Only if logged in and registered	Only if logged in and registered

Closed	Only if logged in	as an editor for the particular labyrinth	as an editor for the particular labyrinth
Private	Only if logged in as an editor for the particular labyrinth		
Key	Requires a text-based key to run		

Table 1: security settings and their meaning

5.5 Delete

You may wish to delete a labyrinth you have started. To do so click the 'Delete' link on the main menu – you will be challenged whether you really want to delete this map. If you are sure, click the 'go ahead and remove this labyrinth' button. Recognising that some people may wish to resurrect deleted maps, they are not permanently deleted but disabled and held offline. Contact an OpenLabyrinth administrator to reinstate a deleted map or, if you have SQL access, go to the database and change the mapEnabled flag to 'y' in the MAP table.

5.6 Visual Editor

The Visual Editor is a graphical tool that allows you to add, edit, move and delete nodes, links and node content rather like we previously did with the VUE concept mapping tool, but now inline to OpenLabyrinth so that edits are made directly to the labyrinth and can be previewed at once.

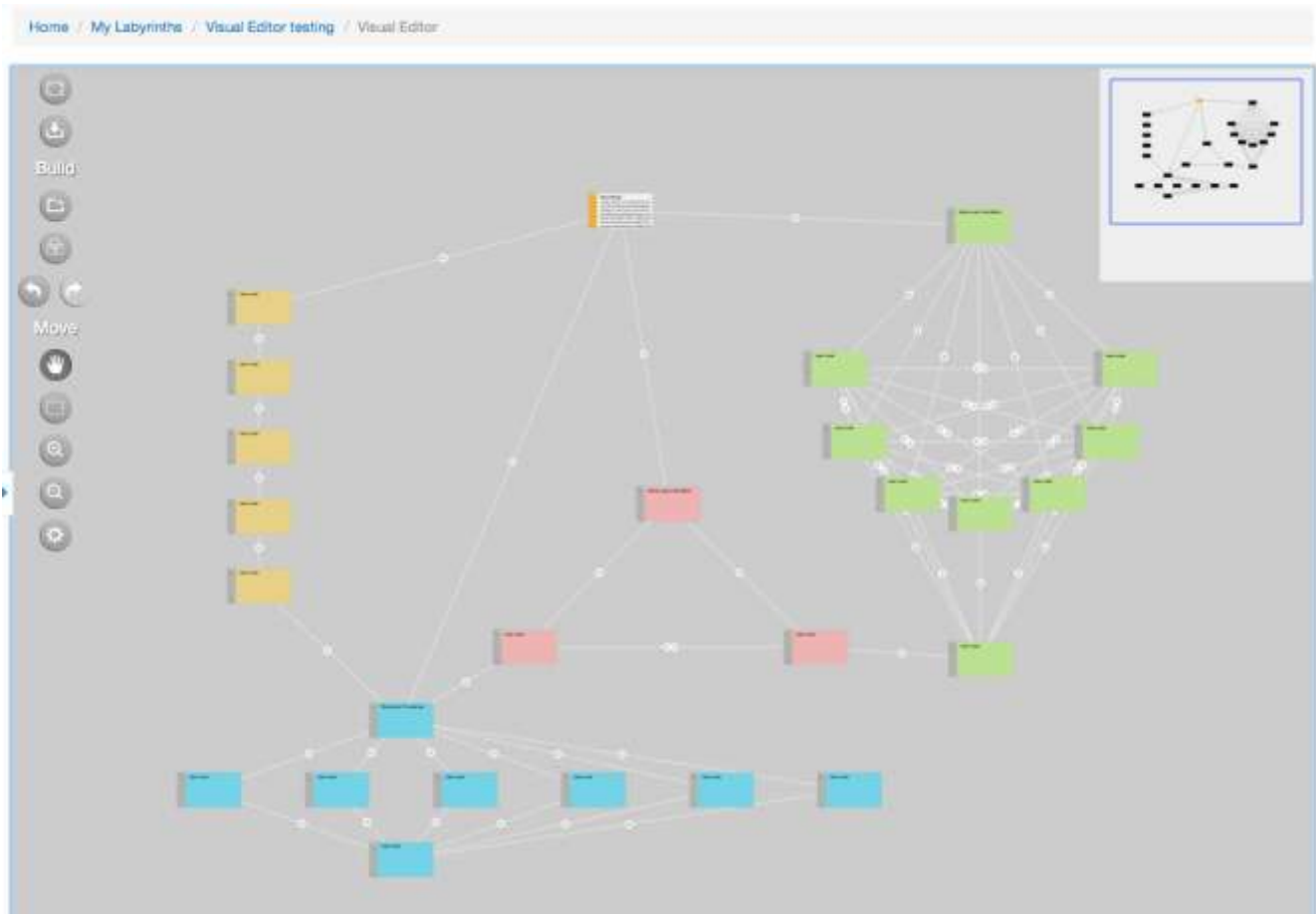









Figure 14: the Visual Editor

	Full screen mode: minimize other OpenLabyrinth elements to maximize the drawing desktop space.
	saving your work: the visual editor updates or saves changes to the database. Quick-key is Alt-S.
	adding node boxes: click on the + icon at the bottom left of a node box to add a new node linked from the one you clicked. Alternatively click the + icon at the top left of the editor to create a new node not linked to another node. Note that the viewer needs to be updated before any changes are added to the database.
	Insert pre-defined template: you can insert a sub-set of nodes. Select from various pattern designs: linear, branched, dandelion
	moving node boxes: click and drag in the left side bar in a node box to move it around the screen. All links to or from the node box will move at the same time..

	editing text in node boxes: click on the title or body text of a node box to start editing it. A popup floating editor panel will allow you to change text, color, title, info, and a number of other node attributes. Remember to Save before moving to the next node or link. You can leave this panel open and it will refresh with the newly selected item's contents. You can drag the panel around the desktop to keep it handy.
	change a node box background colour: click on a node box then select the Actions tab in the floating panel to change the background to selected colour from the colour wheel.
	adding links: click on the target icon on the left hand side of a nodebox to create a new link. Click and drag the target on to the node to be linked and let go. A new link has been created. Note that the viewer needs to be updated before any changes are added to the database.
	editing links: click on the arrow icon in the middle of a link to edit it. Select one of 'Direct', 'Back', 'Dual' or 'Delete' and then click Apply. Note that also add a new Link Label here. This will be the text in the choice provided to you, instead of the title of the target node
	panning (moving around): click and drag in an open area to drag the workspace around - a newly imported labyrinth may need to be dragged into view - first zoom out to find it, drag to the centre and then zoom in.
	Multi-select tool: you can select many nodes at once. Draw a box around them and then move/copy/cut/paste them en masse. Quick keys for Copy/Cut/Paste are Ctrl-C, Ctrl-X, Ctrl-V (not Cmd-C etc)
	zooming: click on the Plus or Minus buttons to zoom in or out - the default is 100%. Quick keys are Alt-plus and Alt-minus. Avoid the use of Ctrl-Plus or Cmd-Plus. This can produce unexpected effects
	Settings: for the Visual Editor. Change auto-save interval.

Please also note the following:

- From version (2.5) VUE map imports include the X, Y and RGB properties of every node. Previous imports and any other node content will not by default have these properties and will be shown in a grid format.
- Updating the viewer will save any changes. It is recommended that this is done regularly, particularly when creating new nodes to ensure they are saved and managed properly. The auto-save is not fully reliable.

5.7 Nodes

To edit the nodes in the current map, click the 'Nodes' link in the main editor.

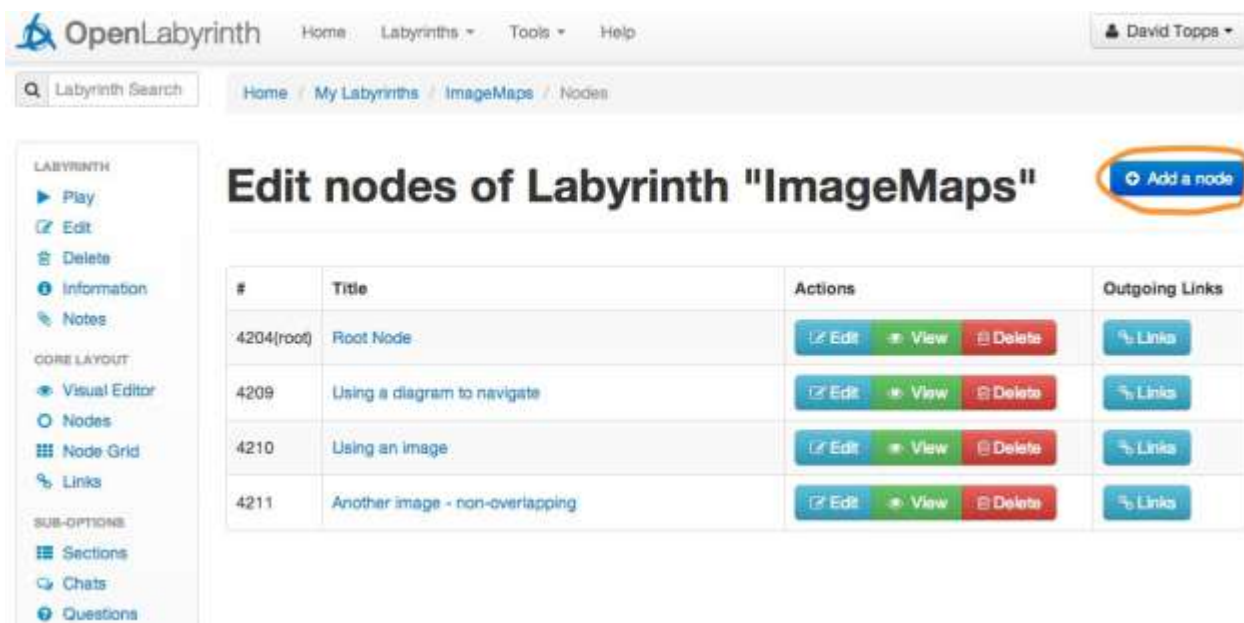


Figure 15: Nodes panel with 'Add a node' button highlighted

This node edit screen lists all of the nodes in the current map. Clicking on the title of a node will preview it. Clicking on 'edit' will launch the node editor and clicking on 'Links' will launch the link editor for that node. There is a button at the top of the node listing page to let you add a new node.

Remember to click on the 'Save changes' button at the bottom of the page, once you have finished editing the details about a node.

Whether you add or edit a node, this will launch the node editor, which has the following features:

- 'Set as Root' button – now at the top of this editor page. Click this to make this node the starting or root node of a labyrinth.
- The first (and default) mode is in WYSIWYG¹ editor that provides an interface somewhat akin to a word processor. In OpenLabyrinth this is provided by the TinyMCE component². There is a button within the editor panel [HTML], which changes you to a direct editing mode, where text and HTML code is entered directly into the content and info fields. This is sometimes useful for fixing small formatting glitches or for inserting other more sophisticated HTML code into the page. Other icon buttons within the TinyMCE editor panel allow you to change text to Bold, Italics, Underline etc, as well as a number of other editing features.



¹ WYSIWYG = What You See Is What You Get, an old term, where the text appears just as you would see it on the page when the labyrinth is played.

² TinyMCE is a platform independent web based Javascript HTML WYSIWYG editor control released as Open Source under LGPL by Moxiecode Systems AB. Copyright © 2003-2006, Moxiecode Systems AB, All rights reserved. For more information visit the TinyMCE website at <http://tinymce.moxiecode.com/>

Figure 16: the WYSIWYG tool bar

- Node title: this is both the title displayed in the lists of nodes and at the top of the page and the default text of any link to the node (although this can be overridden – see section 5.8).
- Node content: This is what is shown to you and is where narrative, images and instructions are added. You can also insert links to Media Resources, Questions, Avatars, Elements and InfoButtons in here using a simple wiki-style of notation. See section [User Interface](#) for more information on how to insert these references. Note that to add a Media Resource such as an image you need first to upload the file and then copy and paste its wiki-style reference into the node content box where you want the file to appear.
- Supporting information: additional content can be added in this field. Any content in the info box for a node will be displayed when you click the InfoButton at runtime. This launches a popup window containing the supporting information. This would typically be used to provide tips, commentary or advice from a character such as a tutor, or links to other materials and references outside OpenLabyrinth.
- Supporting information keyword: this is the name of the wiki-style link that must be inserted into the Node Content window at the point where you want the InfoButton icon to appear in the text. If you do not insert this keyword link then no InfoButton is shown. (This has changed from OpenLabyrinth v2, where you could only have the Supporting Information displayed after all the other node content.)
- Annotation: this field is where case authors can insert hidden markup about a node. For example, for authoring team members to leave each other questions and suggestions. Text in this node is not visible to regular Users of the case. When case Authors are playing the case, such annotations are displayed in a separately marked box on the page.
- Counter functions: for every counter created, this is where they can be dynamically changed. You can leave this blank or use one of '+' or '-' or '=' plus an integer. For instance '+5' adds 5 to the current counter while '=6' sets it to 6 no matter what it was before. See [Counter Grid](#) section for more information on setting counter scores.
- Exit node probability settings: this toggles on off whether just one of the linked nodes will be presented at random or all of them presented at once. This is functionally the same as a Link being 'random – select one'.
- Node Conditional settings: this allows you to set conditional rules for entering the current node when playing the case. These rules specify what nodes need to have been visited before the current one can be accessed. This has largely been superseded by the use of [Rules](#) and will likely be discontinued in future versions.
- Link presentation style: this changes how the linked options are displayed to you:
 - text (default) – each choice is shown as a plain text hyperlink.
 - dropdown – each choice is an option in a drop down list.
 - dropdown + confidence – each choice is an option in a drop down list with a second drop down to let you select how confident they were at making the selection.
 - type in text – user types in what they think the option should be. If, after the first 3 letters are entered, this matches one of the available choices then that choice is selected.
 - buttons – each choice is shown as a button with the destination node as a title

- Node priority: this flags whether the current node must be avoided or must be visited – this is only used in the session reporting and feedback. See section 7 on [Feedback and Reporting](#). It does not affect the user’s navigation of a case. This is not the same as ‘Conditional Rules for node’ above
- Prevent revisit: this allows authors to prevent users from going back to a node that they have visited by removing it from the list of links. In any given node scroll down to “Prevent revisit” and select whether this feature should be on or off for this node. Note that this “lockout” is not complete and can be “undone” by the user. All such moves are tracked by OpenLabyrinth.
- Link to end and report from this node: this is off by default. If turned on, a link is added to end the current session and get feedback – see [Feedback and Reporting](#) section for more on this. Note that only logged in users can view such Reports.
- Remember to click the ‘Save changes’ button at the bottom of the page, once you have finished editing the details about a node.

5.8 Links

The links editor in a map lists every node in the left hand column and every link from that node in the right. Every left hand node entry allows you to edit its particular links or preview it. Every linked node entry on the right allows you to go to that particular node’s entry in the left hand column or preview it.

The editor for a specific node’s links allows you to:

- Change the type of linking from the current node: there are three kinds of node linking that can be set here:
 - Ordered: this allows you to specify the order that links will be shown. This ordering is set in the current link editor using numeric dropdowns per link – they evaluate low-high.
 - Random order: this means that the order of the available links from the current node will be randomised every time the node is loaded.
 - Random select one: this means that just one of the available links will be randomly selected – note that this property can also be set in the node editor.
- Edit current links from the current node: this allows you to order links (if linking type has been set to ‘ordered’), delete links or edit them. If you are editing a link the form looks very similar to that for adding new links.
- Add a new link to the current node: this allows you to select a node to link to from a dropdown list of every node in the current map not already linked from the current node. You can also (optionally) set alternative text for the link (the default is the target node’s title) or set a path for an image or icon to be used instead of a textual link.
- Add a counter function to a link – this will only activate when the link is clicked. This is different from the previous OL model of counter functions only being available on the node.

5.9 Finishing a case

If you are logged in, on the OpenLabyrinth home page there is a ‘My OpenLabyrinths’ link from which you can view a list of every labyrinth you have started, behind which there is a user session report that shows what path was taken, how much time elapsed between entering and leaving the node and the varying level of any counters. A histogram of this data is also available. See section 7 for more information on [Feedback and Reporting](#).

6. Editing for Advanced Authors

6.1 Labyrinth Details

We already covered the simple metadata that can be edited for a labyrinth in [Details](#). If you change the user interface to advanced, using the menu 'Help | Advanced UI', you will see a much more detailed panel of metadata about the labyrinth.

6.2 User Interface

There may be a number of different elements on display (see figure 17) including:

- Title: every node has a title, which is typically displayed at the top of the page.
- Node content: although not mandatory almost all labyrinth nodes will have some text content for the user. Typically that will describe the consequences of having made the previous choices of paths and possibly things to consider in making the next choice.
- Linked options: the way to navigate an OpenLabyrinth case is to click on one of the available options here. These will usually show the title of the node you are linking to.
- Review your pathway: a clickable track of your pathway through the labyrinth since you started is available. Click on a link to go back to that point – note that this does not roll back the track of which nodes have been visited. Every action is recorded in a session. This may also affect how some Counters and Rules operate.
- Counters: a simple way to show your progress in a game. Authors can show or hide counters on each node, minimizing distracting information when not needed. There may also be a display of how the counter's value has changed since the last node.
- Map and Node ID: this shows the current labyrinth and node ID number. If there is a problem with a case, the case authors will appreciate you making a note of which node ID number caused your problems.
- Reset link: this ends your present session and restarts you in a new session within the same OpenLabyrinth case.
- A link to OpenLabyrinth home page: clicking the OpenLabyrinth icon returns you to the home page.
- Other graphics, links, tools and text may also be displayed depending on the current skin.

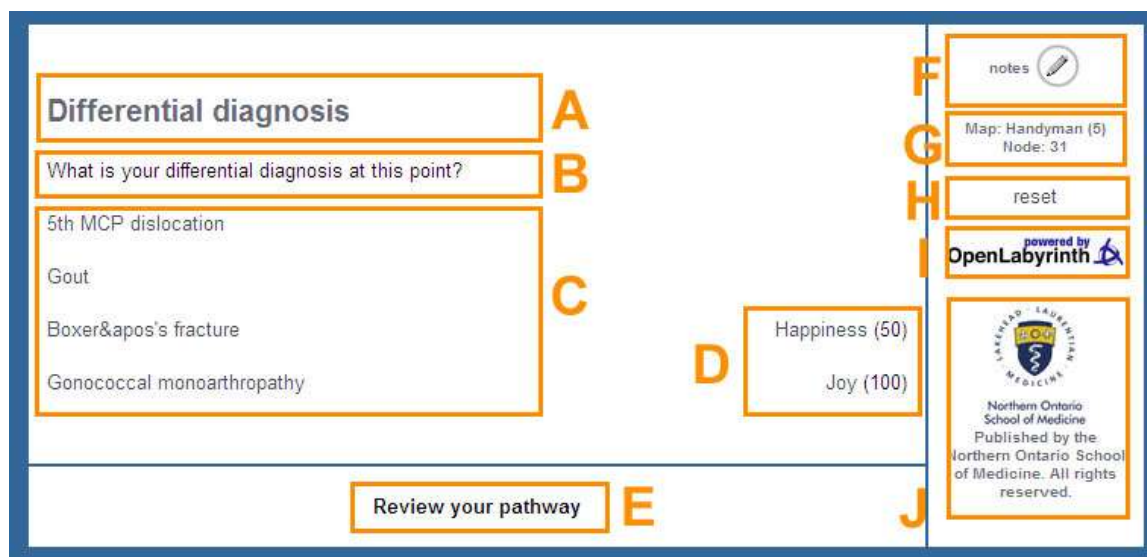


Figure 17: a typical labyrinth screen's elements:

A: Node title; B: Node message/content; C: Links/options; D: Counters – label plus current value; E: Link to review nodes viewed in the current session; F: Open note taker; G: Current labyrinth and node information; H: Reset/restart current labyrinth; I: link to OpenLabyrinth home page; J: Skin-specific graphics and text.

6.3 InfoButtons and context-sensitive Help

There are many ways in which the case author can embed help or hints within a labyrinth. The most common method is with the use of InfoButtons but remember that you can also link out to any page or document on a web server simply by including its URL.

The InfoButton was already described above in [InfoButtons](#). Advanced authors may want to explore further what can be done with the InfoButton panels. For example, although a Node can only contain a single InfoButton panel, you can borrow InfoButton panels from other nodes, either in the same or a different labyrinth on the same server, simply by pointing to its wikiref. This allows you to reuse InfoButtons across cases and keep a common help reference for them. However, it does have the disadvantage that this method is not portable across servers – if you Export your case to a different server, you will need to remap your wikirefs to the new InfoButton numbers. (The easiest way to do this is by using the Replace function in the Node Grid.)

InfoButton secondary panels all have a consistent simplified style similar to a popup window. You cannot change their size or controls. Advanced authors may want to explore what can be done by using hyperlinks that open a new window or tab. Or you can embed simple Javascript in the Node that opens a secondary window, including window size options etc. There are many tutorials on the web that describe how to do this.

You can also call the InfoButton module directly by inserting the same code as direct HTML:

```
<p>2nd node Info: <a href="#" onclick="window.open('/renderLabyrinth/info/2339', 'info', 'toolbar=no,
directories=no, location=no, status=no, menubar=no, resizable=no, scrollbars=yes, width=500,
height=400'); return false;"></a></p>
```

Using the direct HTML approach does allow you to control the size and controls of the secondary window. You can also use a different icon instead of the wee blue “i”.

We have also had good luck with TiddlyWiki, a simple self-contained single-page wiki system. This allows you to create quite a sophisticated, expandable, searchable help file system, with some degree of context-sensitivity.

6.4 ImageMaps and hotspots

One interesting new feature in OpenLabyrinth v3 is the ability to directly enable hotspots in an image. This is not a new web technique at all, but previously it was a bit fiddly to do in OpenLabyrinth. The Node Editor and specifically, the TinyMCE editor, now directly allows you to create an area within an image that has an embedded hyperlink.

Note that this technique is somewhat dependent on screen size and resolution. It does not translate well to small devices such as smartphones so bear this in mind when designing your case. We generally use an image size of 1024x768 pixels (also known as XGA resolution) because this is the standard screen resolution of a data projector or iPad. It still works on many screen sizes that are larger than this.

First you must use the **Node Editor** to do this.

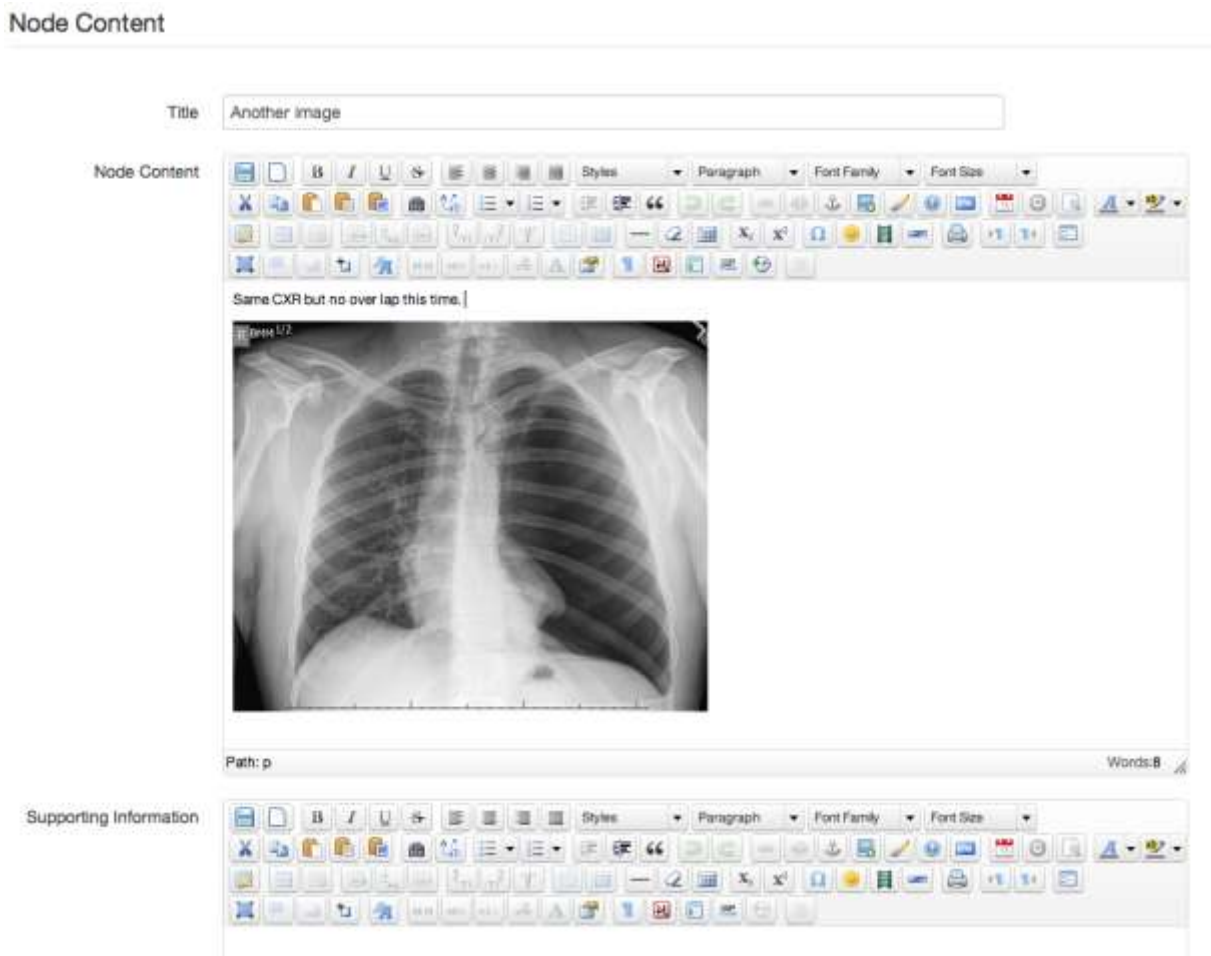


Figure 18: node editor page with a linked image

Normally when you are working with an image in OpenLabyrinth, you would insert the wiki style reference, like [[MR:123]]. But in this case, we need to tell the TinyMCE editor that is part of the node editor more about the picture you are inserting.

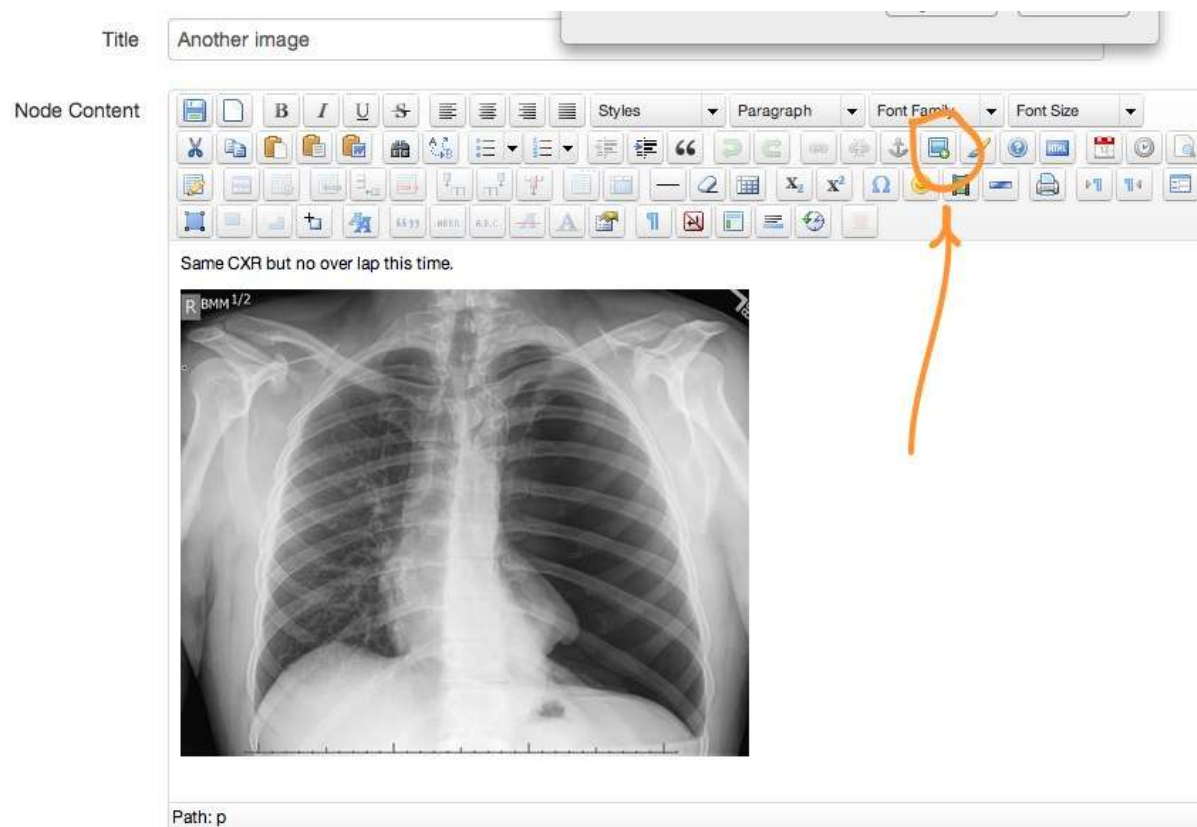


Figure 19: node editor page showing button to insert an image

First insert your image, using the image button. Provide a URL to the image. This is easy for images elsewhere on the web. For images in the Files section for the case, you can still do this but you need to insert the local link which will be something like 'files/imageNameHere.jpg' – if you have done it correctly, you will be able to see the picture itself in the TinyMCE editor. Next click on the 'Image Map Editor' button – see below

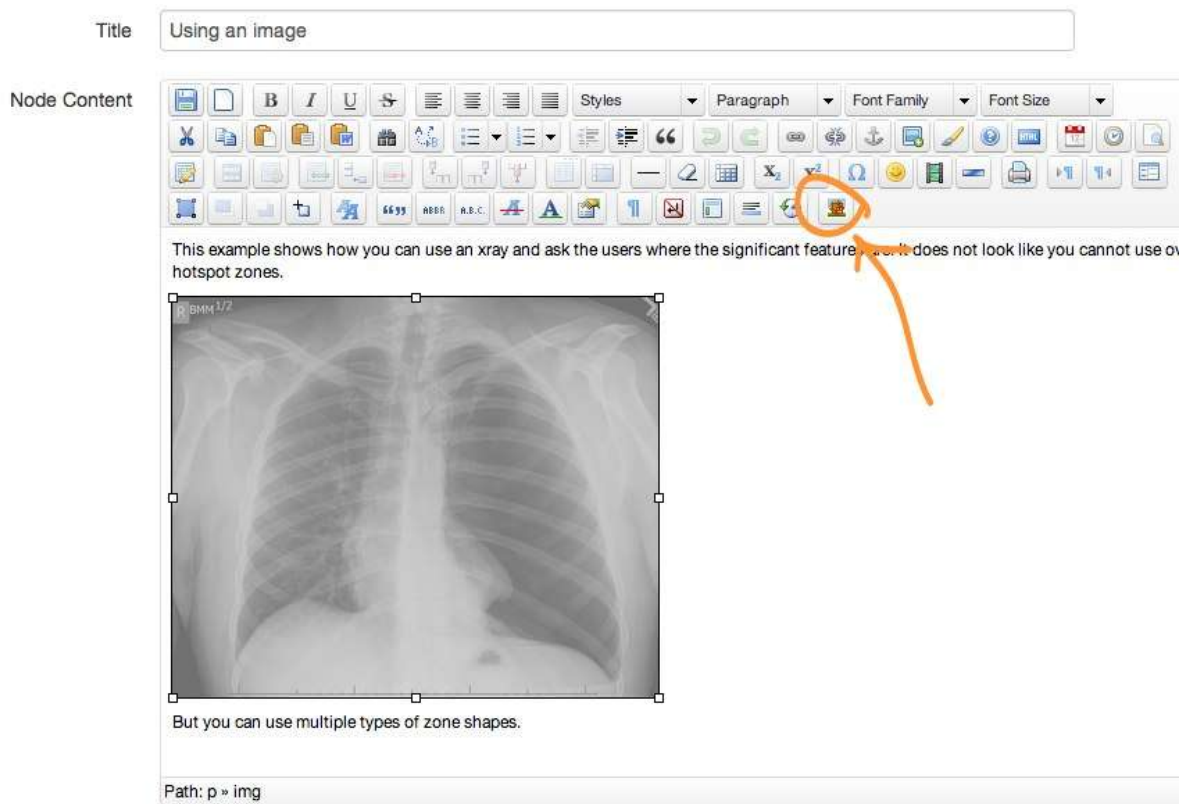


Figure 20: select the image in Node Editor then click ImageMap button shown

This will bring up the editing panel, with your picture in it. You can draw areas of interest, and attach a URL to that shape.

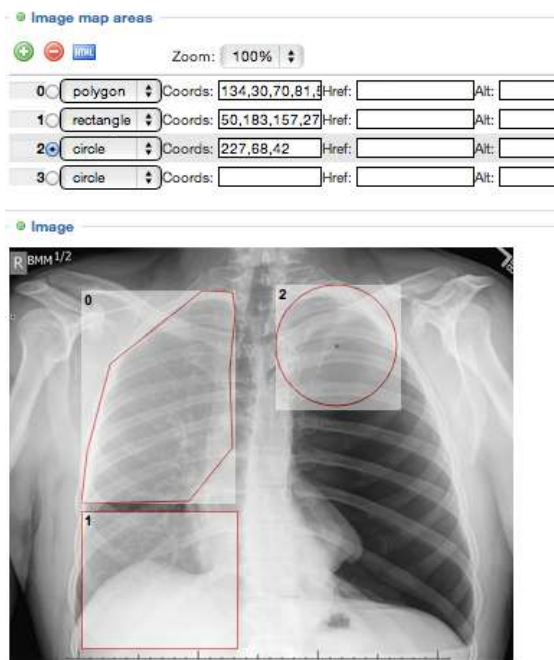


Figure 21: areas can be circles, rectangles or polygons

Simply outline the areas that you want to be active hyperlinks and then select the URL that you wish OpenLabyrinth to jump to. This can be another node in the case or outside of OpenLabyrinth entirely.

Remember to save when you are done.

With this technique, you can make key areas on your x-ray into teaching points – “spot the fracture” etc.

6.5 Node Grid

Sometimes you just want to edit the content of all of the nodes in one go. That is when you use ‘Node Grid’. Basically it provides a basic text box editor for the title and body of every node in a labyrinth for every node in a labyrinth. Be aware that you are working with the whole case at once. This is important when you are part of a team that is collaborating on a case. See [Collaborative editing as a team](#).

The Node Grid has some powerful new editing functions that are helpful in larger cases.

Logic sort:
 On
Off

Find:
← Previous
→ Next

Replace:
Replace
Replace All

Node ID	Title	Text	Info	X	Y	Section ID
4186 (root)	Root Node	Ok, so this has nothing to do with parliament. This is just to illustrate what you can now do with	Why are Q uestions spelled this way?: In <i>this</i> example, it is only because they are referred to in the	479	376	
4187	Calculate my scores	Check out the Scores Counter. Did the Q U estions assign the right		482	600	
4188	Hidden answers	This Question is more secretive. No feedback is provided as to whether your choice is correct or not...		780	553	
4190	Finish case	All done. How did you do? Check out your		843	795	
4755	More questions to test you	With these radio buttons, they have been set up so that users can make multiple attempts at their answers.		105	596	

Save changes

Figure 22: Node Grid editor works like a table

By default, Nodes are still displayed in the order that they were created. In large labyrinths, it can be hard to find a specific node. You can now sort the table by 'Title' to help with this. We have also provided some tools that help you to sort this table according to the logical flow of the case. The 'Logic sort' button will make an attempt to sort the nodes by their logical flow – this works best with cases that are mostly linear in style.

Because most authors will use the Visual Editor as a means to map out the logical flow of a labyrinth, often working from left to right or top to bottom on their visual layout canvas, we have also provided the means to sort the nodes by their x and y positions on the Visual Editor map. (To change these, you must drag the Nodes using the Visual Editor – they cannot be edited within the Node Grid.) For simple cases, authors will not find this necessary, but for very large or complex maps, we have found this most useful.

Sections are another way of logically grouping cases. See 6.6 below. If you have assigned Nodes to Sections, you can also sort the Node Grid according to these Sections.

For making global changes to the text of a case, for example, changing a patient name, there is now a Find and Replace function in the Node Grid.

Remember to Save changes before leaving the Node Grid editor.

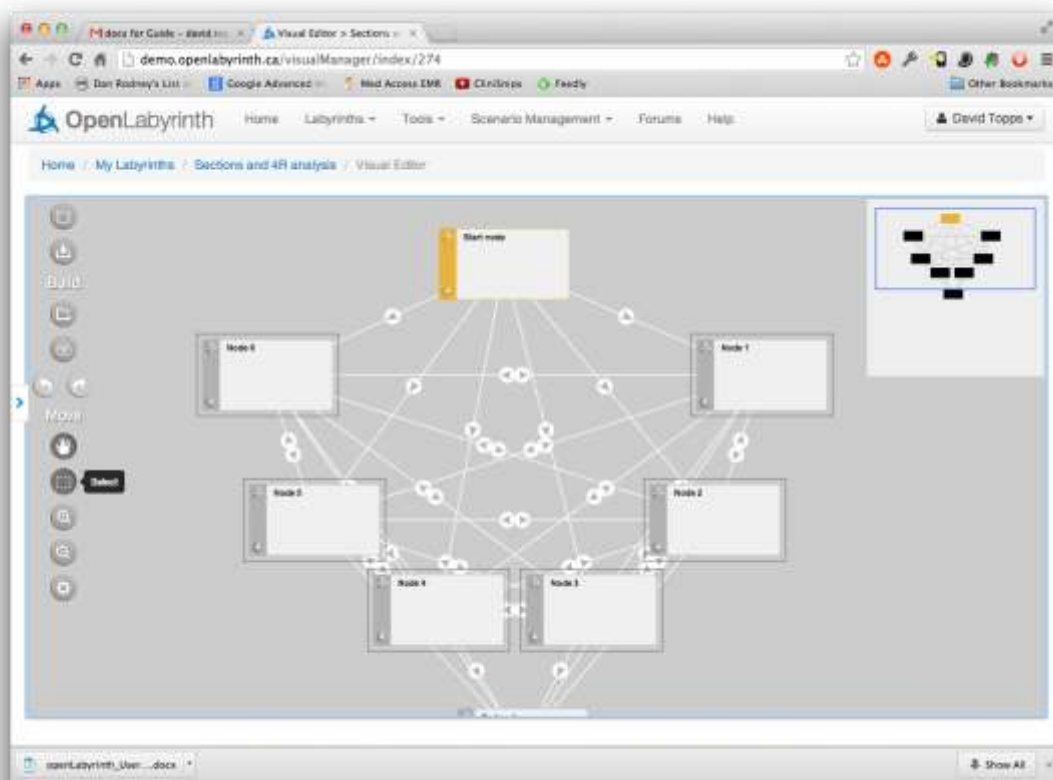
6.6 Sections

Sections were first designed to be simple chapters within OpenLabyrinth. But they have come to be useful in many ways for grouping sets of Nodes together for various purposes:

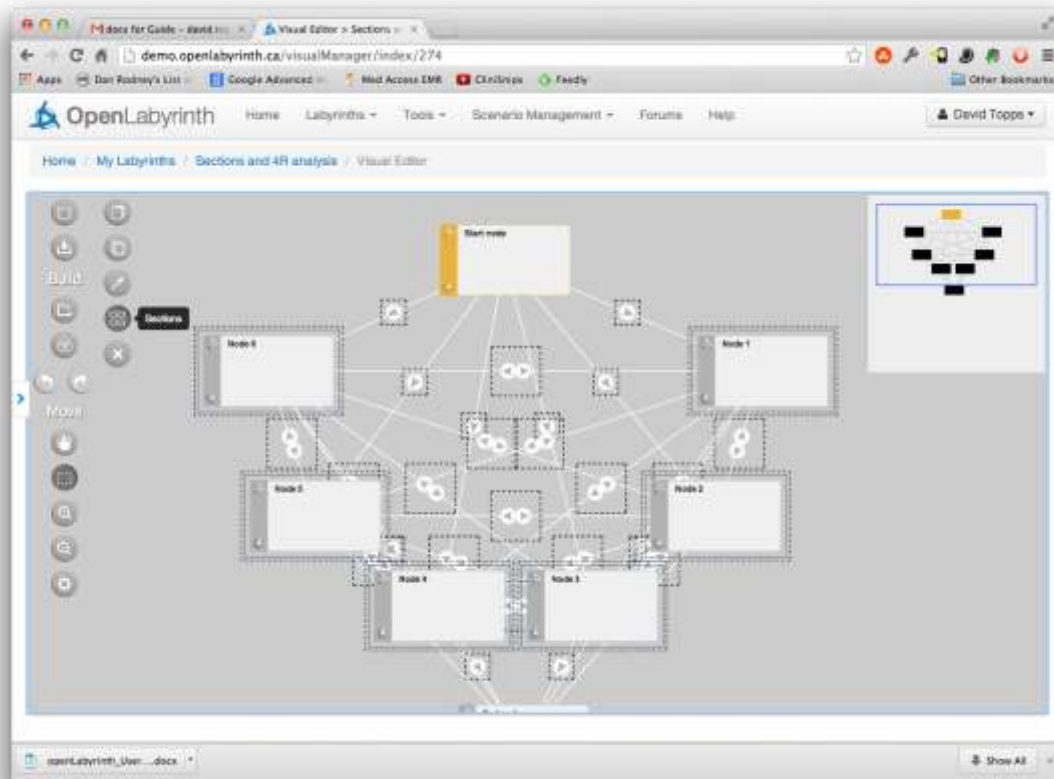
- A navigational grouping, with shortcuts to the starting points
- A set of choices, typically a dandelion, with reporting on order of preference
- A group of nodes to be listed in a set order in Link choices
- A group of nodes to be navigated within a set time interval

Node sections are a way of organising nodes into logical groups. They can be used to help authoring, particularly where there are a great many nodes or several nodes have the same title. They can also be used to create a menu to help users browse through a labyrinth.

Sections can still be created manually with the Section editor on the left-side authoring panel. However, it is much easier for most authors to do this with the Visual Editor, as shown below.



- Click on the Select button on the left side of Visual Editor.
- Draw a box around the group of nodes you want to include in the Section
- Click on the Sections button on the left side of Visual Editor



You can now edit the Sections popup, and change the name of the Section, order of nodes within the Section etc. You will now see that the Nodes that have been included in a Section are now outlined with a second frame of a different color.

If you would prefer to directly use the Section editor on the left-side authoring panel on the main editor page:

- You can add and edit Sections and assign Nodes to Sections
- You can add a Section by typing a label into the form at the bottom of the Section page and clicking submit.
- You can edit an existing Section by clicking on the 'edit' link next to its name on the Section page (which also lists the Nodes assigned to each Section). This provides a list of the current Nodes in the Section that allows you to order them and remove them from the current Section. You can also add any unassigned Node to the current section. Note that a Node can only be assigned to one Section at a time.

You can control how the Sections are shown to you. The choices are: don't show, visible, navigable (clickable). In the latter the section label becomes a hyperlink to the first node in the section.

Sections are very useful in [Rapid Reporting of Real-time Responses \(4R\)](#). When you designate a group of Nodes as a Section, the 4R report will group that set of nodes and show you which nodes were chosen and in what order by the Users. Typically, this is done with a dandelion set of Nodes.

Links are generally presented in a random order for Users to choose from. But sometimes, you want the Links to appear in a certain order. This order can be manually edited using the Links editor. There is now a faster way to create such an ordered list of Links.

Once again, you use the Visual Editor and the same approach to group a set of Nodes into a Section. But this time, arrange your Nodes on the canvas such that they have a descending order in either the x or y-axis.

[insert graphic with a crescent of Nodes]

Click on the XXX button to sort your Links by this order.

You can also use the above quick method to define a Section when you want to highlight a group of nodes within a labyrinth that should be navigated within a specified time frame i.e.a [Pop-up messages](#).

6.7 Chats

These are similar to Questions in appearance. They are simple way of providing extra information to you, if you click on them. They can also control a Counter, so that you can track whether the User clicked on them or not, or control later navigation of the case.

START

This case is a simple illustration on how you can use the Chats feature in OpenLabyrinth.

Chats were originally introduced in OLab2 as a simple way to soliciting more history features. However there are other ways to use them.

Click for more info

Second chat item within the same Chat reference. We have made this second prompt much longer to simply demonstrate that the text within it will wrap to the next line if necessary.

Figure 23: node with 2 Chats before they are clicked

START

This case is a simple illustration on how you can use the Chats feature in OpenLabyrinth.

Chats were originally introduced in OLab2 as a simple way to soliciting more history features. However there are other ways to use them.

Click for more info

This is what you see upon click...

Second chat item within the same Chat reference. We have made this second prompt much longer to simply demonstrate that the text within it will wrap to the next line if necessary.

This is also a much longer response, again to demonstrate that the text within it can simply wrap to subsequent lines if necessary. *Italics*, Underline, and other simple HTML text formatting can also be included in the Response field.

Figure 24: same node with 2 Chats after each is clicked

We have an example case at <http://demo.openlabyrinth.ca/renderLabyrinth/index/277> where you can explore this further.

6.8 Questions

You can now create several kinds of questions:

- Single-line text entry: Free-text questions have no right or wrong answer but you can perform simple text or language processing using [Rules](#).
- Multi-line text entry: this provides a text box where the user can input many lines or even pages of response. The contents of the box will scroll if needed.
- Multiple-choice questions can have right, wrong or neutral answers. Can have multiple valid selections (sometimes known as check boxes). The score from any given question can be used to control a Counter, which can in turn be used to control game play within a labyrinth.
- Pick choice question - only one choice is valid (sometimes known as radio buttons). Each response can also control a Counter.
- Slider question: the user can slide the control button along the scale. This is useful for data that may not have discrete quanta. Can be attached to a Counter.
- Drag & drop question: the user can drag items in the list into a different order of preference. These can be evaluated using [Rules](#).
- Script Concordance Testing: this is a special type of pick-choice question, which is scored and reported differently. It is outside the scope of this User Guide to explicate SCT in detail. Author groups who are interested in exploring this area are encouraged to contact us directly – we are very interested in collaborating in this area.
- Situational Judgement Testing: this is a special type of drag-drop question, which is scored and reported differently. It is outside the scope of this User Guide to explicate SJT in detail. Author groups who are interested in exploring this area are encouraged to contact us directly – we are very interested in collaborating in this area.
- Cumulative question: this is used along with cumulative cases, where the responses are aggregated into a single set, for group use in a Scenario. It is outside the scope of this User Guide to explicate group polling in detail. Author groups who are interested in exploring this area are encouraged to contact us directly – we are very interested in collaborating in this area.

Now for a bit more detail about each of these question types. You may also want to refer to [Rules](#) and [Question Rules](#) for more information on how to make best use of this powerful feature of OpenLabyrinth.

Free-text questions, both single-line and multi-line, generally provide simple recording of text input. Data entered here will be stored in a Counter value – the name corresponds to the Question ID number (e.g. [[QU:1234]] will be stored as [[QU_ANSWER:1234]]) that can then be used in [Rules](#).

You can also apply a Validator string – this is most useful for single-line text questions. This allows you to specify letters, numbers or various combinations of these. We have made use of the excellent third-party

Javascript validator tool – see <https://github.com/chriso/validator.js> for more information about this tool. Also see [Text Validators](#) for more details on how to use this validator tool.

We now have limited text-matching so that OpenLabyrinth can now interpret the user response. Note that this is fairly crude and we caution authors not to expect anything like Natural Language Processing – we are a long way off that. See [Rules](#) for more information on how to use the syntax and keywords available. We encourage others to explore what can be done with this and may create an area on our web site where authors can post neat examples of what can be done with this, if there is interest.

We have provided this text-matching as a means to provide increased flexibility for case authors who do not wish users' choices to be completely pre-determined. We have moved beyond the older design which used [Key Feature Problems and Matching](#) in OpenLabyrinth v2. However, Natural Language Processing is difficult to get right and easy to get wrong, leading to frustration for both players and authors.

Multiple choice and pick choice options can be stacked vertically or horizontally across the page. Be careful with horizontal stacking because the response feedback is displayed by the choice, which can spoil your layout.

These are created in the 'Questions' part of a labyrinth's editor and, once created, a `[[QU:xxx]]` tag is created. Copy and paste this into a Nodes Content box to render the question at runtime, just as you might for a Media Resource/image `[[MR:nn]]` tag. Multiple questions can be placed on a Node page.

Questions for "Question Time"

[+ Add Question](#)[Paste question](#)

Embeddable	Stem	Type	Actions
<code>[[QU:97]]</code>	Single line text question	single line text entry	Edit Duplicate Delete
<code>[[QU:98]]</code>	Multi-line text entry	multi-line text entry	Edit Duplicate Delete
<code>[[QU:99]]</code>	MCQ with best of 5	multiple choice	Edit Duplicate Delete
<code>[[QU:100]]</code>	Pick choice style uses check boxes	pick choice	Edit Duplicate Delete
<code>[[QU:101]]</code>	What is the likely diagnosis?	multiple choice	Edit Duplicate Delete
<code>[[QU:104]]</code>	How many responses allowed?	multiple choice	Edit Duplicate Delete

Figure 25: the Questions editor

After clicking on the blue 'Add Question' button and choosing which style of Question to create, you will then see a page like this:

Question type:

Stem:

Show answer to user:

Layout of answers:

Show submit button:

Track score with existing counter:

Number of tries allowed:

Feedback:

Figure 26: starting off in creating a Question

Most of the fields are pretty self-explanatory. You will note that it is pretty easy to flip back and forth between 'Multiple choice', 'Pick choice' and 'SCT' questions. You can choose not to give any indication of whether a choice is correct, either by marking the response as Neutral, or by selecting 'Do not show answer to user'.

The additional 'Show submit button' option is a way to force OpenLabyrinth to process the response immediately, perhaps to show Counter scores or provide specific instructions. If there is no Submit button, the Question's responses are evaluated when the user leaves this nodes and moves on to another.

You can choose whether to allow the user to have multiple tries, possibly receiving feedback on their choices.

You can add as many responses as you wish, using the 'Add response' button. (Previously in OpenLabyrinth v2, you were limited to 2,3, 5 or 9 part questions.)

Slider type questions are edited in a slightly different manner:

Min value

Max value

Default value

Step value

Stem

Orientation horizontal vertical

Show/hide chosen value Show Hide

Skins or appearances Default

Ability to directly input value Yes No

Counter no counter

➕ Add response Save changes

Figure 27: editing a Slider type question

You can specify the minimum, maximum and default values for the slider's scale and where the pointer lies when it is first displayed. The step value controls how finely the range is broken up. The scale can be vertical or horizontal. You can also choose whether to display the actual value stored, in a numeric field, beside the slider, and whether to allow the user to enter a number into that field directly.

There are a number of slider skins or appearance styles to choose from – play with these to find the style you like best.

You can also add Responses to a slider Question. This is a way of converting a value returned by the slider into a predefined Counter score, according to a set of ranges that are defined by the labyrinth author. If no Response is added, then the slider value will simply be stored in the Counter, if defined.

If you have more than one Question per page, all of the responses selected will affect the Counter scores separately. But also note that the same Question cannot be used multiple times on the same page – the

browser will treat them as identical items – later instances will mirror earlier instances, and vice versa. Note that the answers for all questions are provided in the end of session Report.

For some authors, we know that using Questions will be popular and that oftentimes, the Questions will only be slightly different from one situation to another. To decrease the tediousness of recreating many similar Questions, we have provided two methods. Within the same case, you can simply Duplicate a Question by clicking on the grey [Duplicate] button in the Questions page. You then edit the duplicate question, changing responses or counters as desired.

If you want to copy/paste a Question from one labyrinth to another, it is only slightly more involved. First make a note of the Question ID number that you wish to use for your source example. So, if you want to use `[[QU:1234]]` in another labyrinth, do not simply paste that wiki-style link into that labyrinth like you would with an image `[[MR:23]]` – this will not work reliably. Go to the destination labyrinth where you want to insert the question and in the Questions editor page, click on the grey ‘Paste question’ button beside the blue ‘Add question’ button. This will pop up a small dialog like this:

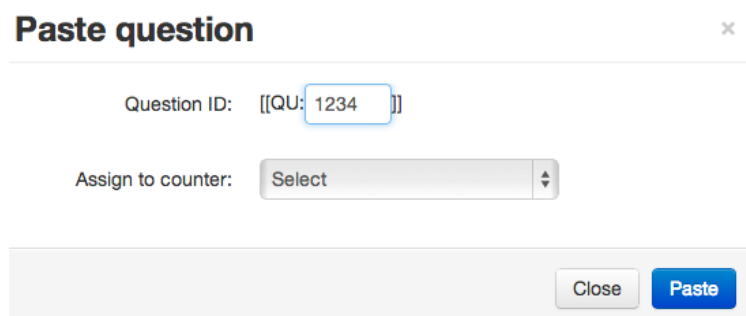


Figure 28: inserting a Question into another labyrinth

Simply enter the number of the Question (you do not need the wiki style square brackets surrounding the number – these are attached for you). The details and Rules from the original Question will be imported into the current case for you. If you want to assign the scores to a Counter, select one here. Click the blue ‘Paste’ button to insert a copy of the original Question into the current labyrinth. You can edit this Question in the usual manner thereafter.

6.9 Question Rules

6.10 In a manner similar to the more general Rules that affect Counters, you can also create small Rules Questions. The syntax is very similar and generally uses the same keywords. For some things, they to use because you do not usually have to deal with the ‘Text Validators

You can use the Validator field on free-text input to limit what the user can enter into the field.

For a full list, consult <https://github.com/chriso/validator.js/blob/master/README.md>

- **equals(str, comparison)** - check if the string matches the comparison.
- **contains(str, seed)** - check if the string contains the seed.
- **matches(str, pattern [, modifiers])** - check if string matches the pattern. Either `matches('foo', /foo/i)` Or `matches('foo', 'foo', 'i')`.

- **isEmail(str)** - check if the string is an email.
- **isURL(str [, options])** - check if the string is an URL. options is an object which defaults to {
protocols: ['http', 'https', 'ftp'], require_tld: true, require_protocol: false,
allow_underscores: false }.
- **isAlpha(str)** - check if the string contains only letters (a-zA-Z).
- **isNumeric(str)** - check if the string contains only numbers.
- **isAlphanumeric(str)** - check if the string contains only letters and numbers.
- **isLowercase(str)** - check if the string is lowercase.
- **isUppercase(str)** - check if the string is uppercase.
- **isInt(str)** - check if the string is an integer.
- **isFloat(str)** - check if the string is a float.
- **isNull(str)** - check if the string is null.
- **isDate(str)** - check if the string is a date.
- **isAfter(str [, date])** - check if the string is a date that's after the specified date (defaults to now).
- **isBefore(str [, date])** - check if the string is a date that's before the specified date.
- **isCreditCard(str)** - check if the string is a credit card.

For example, the following Question has an email validator. Note that it only checks from the correct format of an email address. It does not check whether the email identity itself is real and valid.

Stem	<input type="text" value="What is your email address?"/>
Width	<input type="text" value="50"/>
Prompt text <small>Text will automatically appear in response area. Use to give learner a hint or further instruction.</small>	<input type="text" value="press Enter to validate"/>
Rule	<input type="text"/>
Show submit button	<input type="button" value="Show"/> <input checked="" type="button" value="Do not show"/>
Private	<input type="checkbox"/>
Used	<input type="text" value="1"/>
Validator	<input type="text" value="isEmail"/>
Validator error message	<input type="text" value="This is not a valid email address"/>

Double loop problem¹, as explained in the Appendix.

Question Rules are most often used with single line text entry Questions. Here is a typical example of such a Question being edited, with a crude example Rule:

Edit question "Does this question work?"

Stem

Width

Prompt text
Text will automatically appear in response area. Use to give learner a hint or further instruction.

Rule

Show submit button

Status The rule is correct.

Figure 29: editing a single line text entry Question and Rule

The commonest use of such Question Rules is to change a Counter value or to force the User to move to a different Node. Note that these rules are evaluated by OpenLabyrinth when the User changes nodes, not when the data is entered into the Question and not when the User clicks the [Submit] button attached to the Question.

For a more detailed explanation of how Rules work, see 6.14 below

6.11 Avatars

Avatars are animated simple images of the characters you might want to include in your labyrinth either as passive representations of the characters in your narrative or taking part via speech or thought bubbles. There can be any number of avatars per labyrinth – for instance the same character in different settings and saying or thinking different things or perhaps many different interacting characters.

Each avatar can be customized as follows: Sex, age, eyes open or closed, outfit, mouth shape, the colour of their outfit (for some outfits only), their nose type, hair type and colour, three different layers of accessories such as wounds, glasses, bandages etc., skin tone and colour, background colour, background scenery, background weather and the kind of speech or thought bubble used and what text should go in it.

Once created each avatar has a reference that looks like `[[AV:123]]`. Copying this tag and pasting it into the contents of a node or info box and allows it to be incorporated and reused anywhere in the current Labyrinth. You can duplicate an existing avatar for instance to create a series around a single character, and you can edit them and delete them. See figure 30.

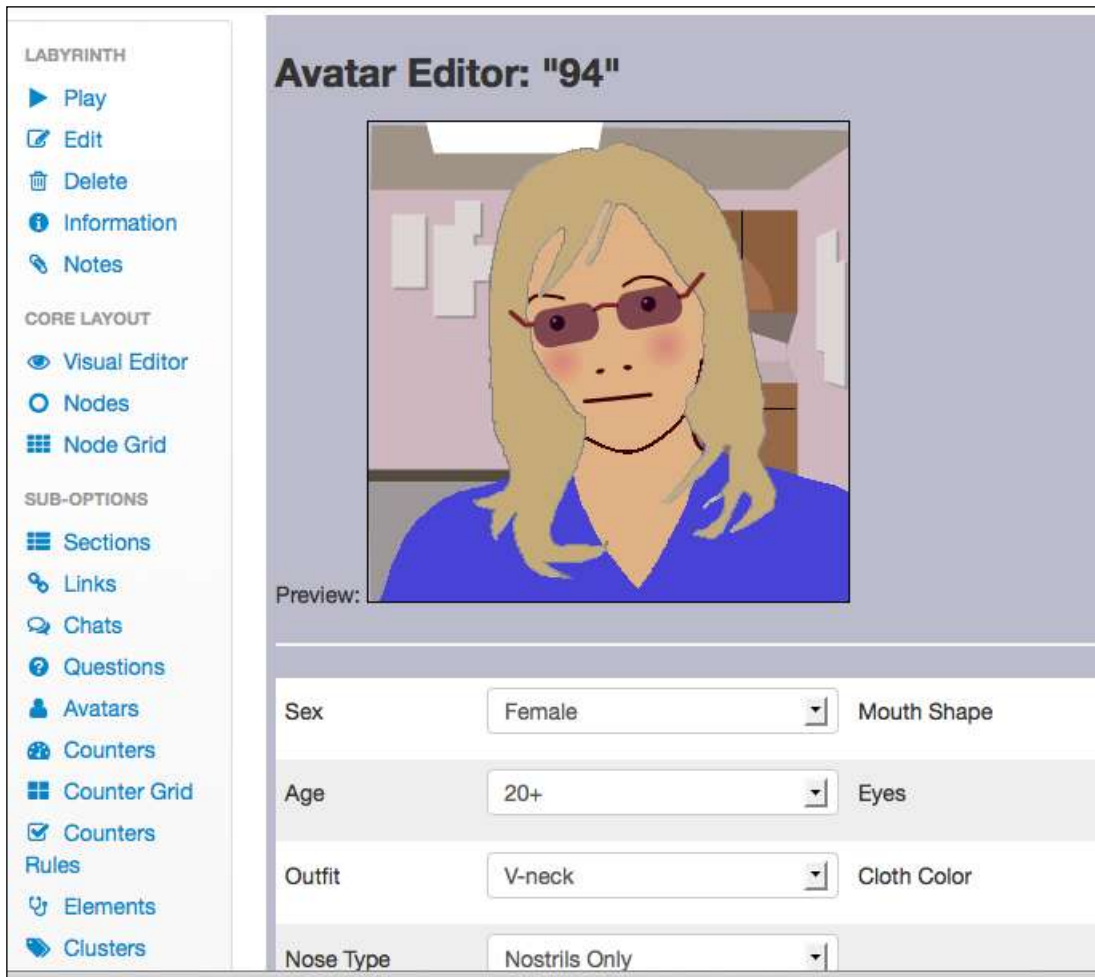


Figure 30: the avatar editing screen

In OpenLabyrinth v3, these are created using HTML5. We decided to stop using Adobe Flash because a number of major software vendors like Apple decided to stop supporting it on their devices. Apple iOS mobile devices do not support Flash. Frustratingly, Microsoft has been poor at adopting HTML5 – neither IE7 nor IE8 supports HTML5 properly. However, Windows users have an easy workaround and can install a much better browser for free. We recommend the use of Firefox or Chrome for running our cases.

6.12 Counters

Counters are comprised of a label and a value, which can be dynamically changed as you work through a labyrinth. There can be any number of counters and each of these counters can have any number of rules that can be triggered by a counter's value. Values can now be integers, floating point or strings.

Counters need to be created globally for a map. To create a new counter or edit an existing one click the 'Counters' link. This opens the counters editor, which lists each existing counter with links to edit, preview or delete it along with a link to create a new counter. Adding or editing a counter opens the counter editor window (see figure 38).

The counter editor allows you to define/change:

- Counter title: this is the label shown for the counter.

- Counter description: an optional text description of what the counter is and what its purpose is.
- Path for a counter icon: an optional path for a graphical icon for this counter.
- The starting value for the counter

Once created, each counter will be displayed (except on some skins) to you indicating its title and current value. If the value is changed at a particular node that value change is also displayed. Clicking on a counter link will launch a popup window with full details about that counter. Changing the value of a counter as you moves through a map is set in the node editor which has a function box per counter: you can leave this blank or use one of '+' or '-' or '=' plus an integer. For instance '+5' adds 5 to the current counter while '=6' sets it to 6 no matter what it was before.

You can now embed Counter values directly within the Node's text content, if you wish. Simply place a tag, as you would for media tags `[[MR:nn]]`, Elements `[[VPD:nn]]` and InfoButtons `[[INFO:nn]]` but in this case, you use the tag `[[CR:nn]]`, just as you would in a Counter Rule definition. See Figure 34 an embedded data element in the node editor (left) and how it renders on screen (right)

6.13 Counter Grid

Counter Grid shows a function box for every node and every counter so that counter functions can be set all in one go. Counter visibility can also be set on or off, for each node in the grid, or all on, or all off.

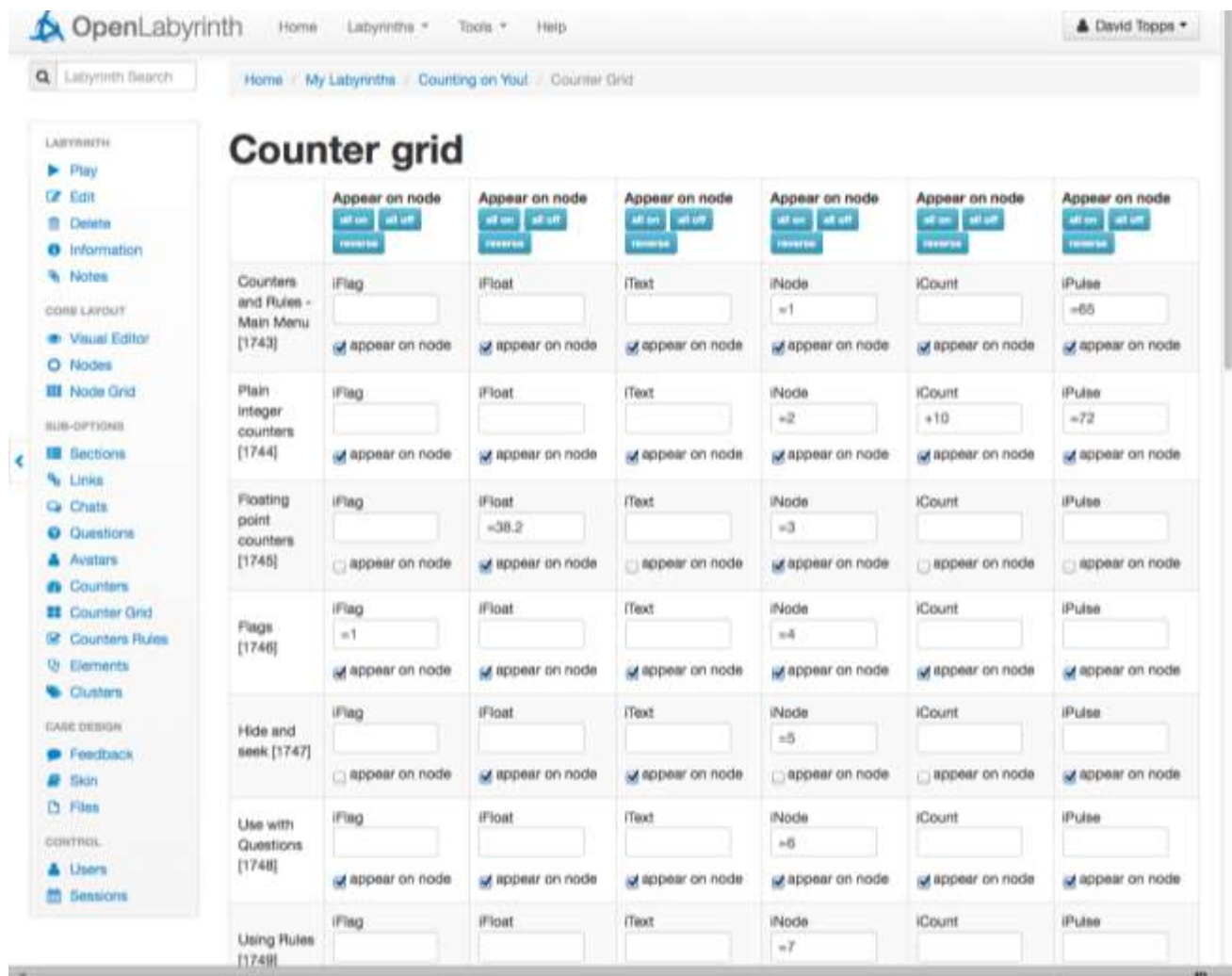


Figure 31: Counter Grid with six Counters

Be aware that this tool affects the case widely. This is important if you are collaborating with team members when writing a case. See [Collaborative editing as a team](#).

6.14 Counter displays

Counter values can be displayed on each node, and these can be turned on or off for each counter on each node. You can also directly refer to the value of a Counter within the text of a node by using its wikiref e.g. [[CR:1234]], and the Counter value will be inserted into the node's text. It will vary according to the current value of the Counter.

We have also developed a method to embed the Counter's value within an image, somewhat similar to a Skin.

Here is a glucometer display - it should show 18.2 mmol/l.

Glucose



The value displayed will match that of the current Counter's value.

You can choose a font from the series of publicly available fonts at

<https://www.google.com/fonts>

without having to worry about font licensing.

Figure 32: screenshot of Node with embedded Counter Display

There is a fine degree of control over the exact placement of the Counter value within the image, including tilt within the frame so that you can align the numbers with the image.

To create a new Counter Display, click on Counter Displays on the left side editing menu within the case. On the next page, click on the blue [Add a display] button in the top right corner.

There are 3 tabs on the Counter Display editor. First set up the frame or panel that surrounds the Counter Display. It helps if you know the size of the image you are working with.




Panels Images Counters

Create Panel Width: 207 px Z-index: 0 Border: 0 Border radius: 0 px
 Height: 368 px Angle: 0 Border color: Background color:

On the second tab, you can add an image file. You can also tilt the image within the panel or frame.

Panels Images Counters

+ Add files... Z-index:
 Angle:

 delete  delete  delete

You can see from the above figure that there are a number of images to choose from. Any image that you have added for this or any other Counter Display can be used – there is a common collection of Counter Display images so that you can reuse them across other cases on that server. Select the image you want from this group.

On the third tab, you can edit and place the Counter value on the image.

Panels
Images
Counters

Font name: Atomic Age
Add
Delete

Label font family: Andale Moir
Value font family: Atomic Age

Label font size: Default
Value font size: Default

Label text: Glucose
Value color: rgb(0, 0, 0)

Label color: rgb(0, 0, 0)
Value z-index: 10

Label z-index: 10
Value angle: 0

Label angle: 0
Value font settings: B I U

Label font settings: B I U
cInt Temp saO2 Glucose











You can import an additional font from <https://www.google.com/fonts> - simply insert the font's name and click the [Add] button. Unfortunately, for now, you still need to visit and grab the name from the page <https://www.google.com/fonts>

You can also edit other parameters for the font such as size, color, tilt angle, bold/italic etc.

To place the Counter on the image, simply click its name from the list of Counter in this case and then when it appears in the top left corner of the panel frame, drag it into place on the image. This works in a reasonably WYSISWYG fashion but you may find that it is not entirely accurate so you may need to do some tweaking.

Objects that you add to the Counter Display are also itemized down the right side of the editor. You can hide or delete these from the current set.

Remember to save your edits with the blue [Save changes] button in the bottom right corner of the editor. Once you have saved your changes, you will go back to a list of the current Counter Displays.

ID	Actions
<input type="text" value="[[CD:125]]"/>	 Edit  Delete
<input type="text" value="[[CD:126]]"/>	 Edit  Delete
<input type="text" value="[[CD:127]]"/>	 Edit  Delete
<input type="text" value="[[CD:128]]"/>	 Edit  Delete
<input type="text" value="[[CD:129]]"/>	 Edit  Delete

To use your Counter Display, simply copy and past the identifying wikiref into the node's text:

Here is a glucometer display - it should show [[CR:450]] mmol/l.

[[CD:128]]

The value displayed will match that of the current Counter's value.

You can choose a font from the series of publicly available fonts at <https://www.google.com/fonts> without having to worry about font licensing.

p

Words: 43

See the wikiref [[CD:128]] in the middle of the above node. You will also notice the plainly inserted Counter value [[CR:450]] on the top line of the page. Check out 'Figure 32: screenshot of Node with embedded Counter Display' to see how this displays.

6.15 Rules

Rules, (previously known as Counter Rules) can be added from the editor for a particular counter. Each rule consists of the following components:

- An integer value
- An operator: equal to, not equal to, less than or equal to, less than, greater than or equal to, greater than

- An action if the condition is met:
 - Go to one of the nodes in the current labyrinth. Note that the counter value should be reset at the target node or the program will loop
 - Show a message in place of the default node message

For example: if counter 3 is greater than or equal to 50 then show message “well done”.

From this versions counter functions can also be set on links. This means that the value of a counter can be changed based in the link selected rather than the node entered.

From this version the visibility of each counter can be controlled by setting it to ‘show’ or ‘do not show’.

For more detailed information, see [Syntax for Counter Rules](#).

6.16 Conditional Logic in OpenLabyrinth3

It is not long before case authors want to be able to modify how a labyrinth behaves, depending on what the user is doing. The learner may be more or less skilled, experienced or knowledgeable; the case may be used in different ways; other factors may affect the outcome of the case. A good author can adapt to all these issues by introducing some conditional logic and variability into a case. OpenLabyrinth provides several ways of doing this – our most powerful tool for this lies within the Rules that can be applied to Counters and to QQuestion responses.

Adding these Rules to OpenLabyrinth has opened up many possibilities. While we have tried to keep things simple for our case authors, this has been a work in progress. We welcome suggestions on how to make this easier.

Note that the simple syntax described below has many limitations. OpenLabyrinth does not have its own internal programming language – we leave that to others to extend in future. There is a simple set of IF.. THEN.. ELSE type structures. If you have done any sort of rudimentary programming before, then will not be hard to figure out. However, there are also many gotchas. We will describe some simple steps but if things are not behaving as they should, you may need to dig into the darker corners of the manual or cast us an email query.

In OpenLabyrinth2, we had simple Counters and a few very basic Counter Rules. We have extended this concept to make things more flexible and powerful. See section on [Rules](#) for more information. For the purposes of this section, where we doing some programming-like activity, you can think of Counters as variables.

Basic Syntax

The general syntax and conditional structures that are available in OpenLabyrinth are very basic indeed. Real programmers will be quite disappointed at what is missing. We usually write our keywords in UPPER case.

IF.. THEN.. ELSEIF.. ELSE.. ENDIF

There are a few variations on the above basic structure. Below are some examples that you can follow.

There are simple keywords that affect case or rule navigation:

GOTO = jump to that node.

STOP = stop processing this rule completely in this labyrinth

BREAK = break from any further processing of this rule, until there is a node change

NO-ENTRY = cannot continue into this node

There are few keywords that help with text or number processing:

MATCH = does this word appear in this string of characters?

UPPER = convert these letters to upper case

LOWER = convert these letters to lower case

PROPER = convert these letters to Capitalized case (as in proper names)

MOD = modulus of this number

DIV = integer divisor of this number

QQuestions can also have Rules embedded within them. The syntax for these rules is largely the same. They are a wee bit easier to work with because there is no 'double-loop' problem. We have a few extra keywords that are specific to QQuestion Rules:

QU_ANSWER = the string of characters just entered in response to this Question

NOT-MATCH = does this word not appear in this string of characters?

CORRECT = show the User that the Response is correct

INCORRECT = show the User that the Response is not correct

Variables such as Counters, Node numbers, Question responses, are written inside wiki-reference square brackets, as you would in the text of a Node:

[[CR:123]] = Counter number 123

[[NODE:789]] = Node ID number 789

[[QU_ANSWER]] = the current Question response (only in Question Rules)

In the general Rules equation editor, you have the choice of using Counter names or counter numbers, and Node names or Node numbers. They will be parsed from one into the other for you.

In the QQuestion Rules editor, it is a much simpler tool. You have to use Counter numbers and Node numbers. It is not able to parse from one to the other.

The usual range of operators are available:

+, -, /, * all mean the same here for numbers as with other languages where you want to add, subtract, divide and multiply numbers. + can also be used to concatenate strings.

= sign can be used for testing equality or for assigning values. There is no difference, as there is in some languages. != is used for NOT EQUALS. Otherwise the usual range of comparators is there:

<, <=, >, >= all mean the same thing as they do elsewhere.

For logical Boolean keywords, there is only '**AND**' and '**OR**'.

Rules Editor

Before you can start working with Rules, you will need to assign some Counters to the labyrinth. See [Counters](#) for more information about this.

Click on 'Rules' in the left side editor panel in your case. This will bring up a list of existing Rules:

LABYRINTH

- Play
- Details
- Delete

CORE LAYOUT

- Visual Editor
- Nodes
- Node Grid
- Links

SUB-OPTIONS

- Sections
- Chats
- Questions

Rules for "Counting on You!!"

Add rule

#	Correct	Rule	Actions
4	Yes	IF <code>iFlag</code> =1 AND <code>iNode</code> = 7 THEN GOTO <code>Using Hidden Nodes and Rules</code> , <code>iFlag</code> = 0	<a>Edit <a>Delete
5	Yes	IF <code>iNode</code> = 10 THEN IF MATCH(<code>iText</code> , "YesText") THEN GOTO <code>Successful logic from String</code> ELSE GOTO <code>Using Rules</code> ENDIF ELSEIF <code>iNode</code> = 14 AND <code>iCount</code> = 0 THEN GOTO <code>Using Rules</code> , <code>iCount</code> = 1234	<a>Edit <a>Delete
17	Yes	IF <code>iNode</code> = 12 AND <code>iFloat</code> = 38.2 THEN GOTO <code>Flags</code> , <code>iCount</code> = <code>iFloat</code> - <code>iCount</code> * 0.5	<a>Edit <a>Delete

If there are none, simply click on [Add rule] to start your first one.

Edit Rule

Text of ruleCode of rule

IF `iFlag` =1 AND `iNode` = 7 THEN GOTO `Using Hidden Nodes and Rules`, `iFlag` = 0

IF `iFlag` =1 AND `iNode` = 7 THEN GOTO `Using Hidden Nodes and Rules`, `iFlag` = 0

Status

The rule hasn't been checked.

Save rule

Check rule

In the Rule editor, you will see two tabs. In the 'Text of rule' rule tab, OpenLabyrinth attempts to parse your entries into things that it recognizes.

Edit Rule

Text of rule

Code of rule

```
IF [[CR:34]] =1 AND [[CR:37]] = 7 THEN GOTO [[NODE:1750]], [[CR:34]] = 0
```

IF **IFlag** =1 AND **Inode** = 7 THEN GOTO **Using Hidden Nodes and Rules**, **IFlag** = 0

Status

The rule hasn't been checked.

Save rule

Check rule

In the 'Code of rule' tab, the underlying wiki-ref style variables are used instead. You will find that each tab will be useful in certain situations.

The [Check rule] button does some basic syntax checking. Correct syntax does not guarantee that your Rule will work as you expected, however.

Examples

We have found that the easiest way to learn and figure out how to work with these Rules is by copying from working examples. Here are a few, but you should consult [Syntax for counter Rules](#) for a broader list of these.

```
IF [[CR:1]] = 1 AND [[CR:2]] = 2 OR [[CR:3]] = 3 THEN GOTO [[NODE:1]];
```

```
IF DiceRoll = 1 THEN GOTO Square 1 ELSEIF DiceRoll = 2 THEN GOTO Square 2 ELSE DiceRoll = DiceRoll + 3
```

```
IF [[CR:1]] > 0 THEN [[CR:3]] = [[CR:3]] + 10, NO-ENTRY;
```

For more information, see the Appendix section on [Advanced Programming in OpenLabyrinth](#).

6.17 Pop-up messages

As part of its large variety of ways of providing Users with feedback and prompts on how they are performing, OpenLabyrinth now provides authors with the ability to show popup messages at timed intervals during a case.

These messages can be triggered to appear at a predefined number of seconds, after:

- the start of, and through an entire journey through a labyrinth
- the start of a Section
- the arrival at a Node

If the User has not completed the designated segment of the labyrinth, a popup message or image will be displayed. The duration of that popup display is also defined by the case author.

Multiple concurrent timers can be running within a single labyrinth. However, if they also trigger other actions, such as a jump to another Node or a change to a Counter value, it is possible to write a set of Popup conditions that will interfere with each other. OpenLabyrinth does no crosschecking for such interference – authors will need to bear this in mind themselves.

The Popup editor page is reasonably self-explanatory for most items. But please note the following:

- Annotation field – this is for collaborative editing, as a means to indicate what the purpose of the Popup is.
- The position of the Popup message can be crudely set using the options of Position and Position type. More fine-grained control is not possible at this time.
- Counter values can be set or modified when the Popup is triggered.
- The style and appearance of the Popup can be modified. You can even create an image-only Popup with no associated text.
- We have experimented with animated GIFs to provide countdown Popup messages – this is a work in progress. Examples will be provided when sorted out.
- Under 'Message Timing', you can set (in whole or partial seconds) the countdown interval and how long the Popup appears for.
- You can also set an Action to occur at that point:
 - None = nothing else happens apart from the Popup
 - Node = a target node where the User is redirected to
 - Report = nothing else shows but a note is made in the Session report
- The 'Message Assign' portion is to allocate the scope of the countdown timer
 - Labyrinth = the timer starts as soon as the labyrinth is launched
 - Node = the timer starts as soon as the User arrives on the Node
 - Section = the timer starts as soon as the User arrives in the Sections

The timing of Popup messages is simple in theory but takes quite a bit of testing and practice to get right. Users become very annoyed if they perceive that they have been cut off too soon, or if messages disappear before they have had a chance to read them. This is even more true if you have multiple Popups programmed into a single labyrinth.

6.18 Elements

The MVP model is made up of four components:

- VPD: virtual patient data – functioning as the electronic patient record, a VPD is a collection of VPD elements. Although designed to support clinical data this could just as easily be used to support any reusable data element such as a character name or unit of measure. These are therefore called 'data elements' in OpenLabyrinth.
- MR: media resources – this includes all supporting files including images and documents. An MR Set is a collection of MR elements. Each MR element has properties of name, path, mime type and arguments. Media resources are handled using the OpenLabyrinth file manager.
- AM: activity model – this describes what you can (and cannot) do. Activities are described as a series of nodes containing narrative and educational content with links between them. Both nodes and links

have rules associated with them to enhance the options for educational game play. This maps directly to the node and link aspects of OpenLabyrinth.

- DAM: data availability model – aggregations of VPD and MR elements. DAM contains at least one VPD element or MR element, although more typically it contains many such elements.

In terms of OpenLabyrinth, its activities correspond to the OpenLabyrinth activity itself, MR equates to the file handling capacity of OpenLabyrinth while VPD and DAM functionality have been added as 'data elements' and 'data clusters' respectively. VPD and MR data elements can be linked directly into an activity node or via a DAM aggregate of VPD and MR elements (see figure 33).

OpenLabyrinth can both export MedBiquitous Virtual Patient packages (a zip file with XML and other files inside) from a labyrinth and it can import MedBiquitous Virtual Patient packages to create new labyrinths.

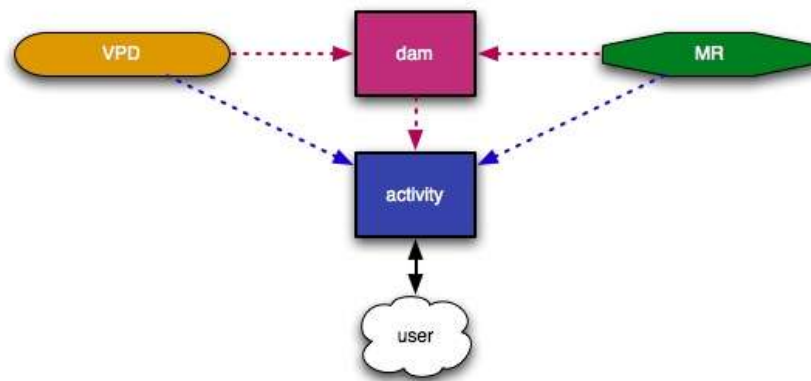


Figure 33: The relationship between VPS, MR, DAM and activity elements in OpenLabyrinth.

The user only works with the activity; all elements are therefore surfaced within the activity.

OpenLabyrinth supports the MedBiquitous Virtual Patient (MVP) data specification, a key aspect of which is the use of data elements. OpenLabyrinth supports the following MVP element types, each of which has a different structure:

- VPDText
- PatientDemographics
- AuthorDiagnoses
- MedicationInterviewItem
- PhysicalExam
- DiagnosticTest
- DifferentialDiagnosis
- Intervention

To use OpenLabyrinth data elements:

1. Select 'data elements' from the editor menu. This will give a list of all the current data elements and a link to create a new element
2. You can edit or delete any of the data elements

3. On the new element page select the type of element you want to create from the drop down menu – this will load a form to be completed to create the element

Using a data element in a labyrinth involves pasting its tag into the content for any given node. The tag has the format of '[[VPD:' then the element's ID and then ']]', i.e. [[VPD:123]] – see Figure 34. VPDtext Elements are sort of like text variables that you can insert multiple times into your labyrinth, for example, to represent the patient's name. Then, if you need to change the name, you only have to change it in one place, rather than going through the case replacing every time it crops up. You can also use data elements in data clusters.

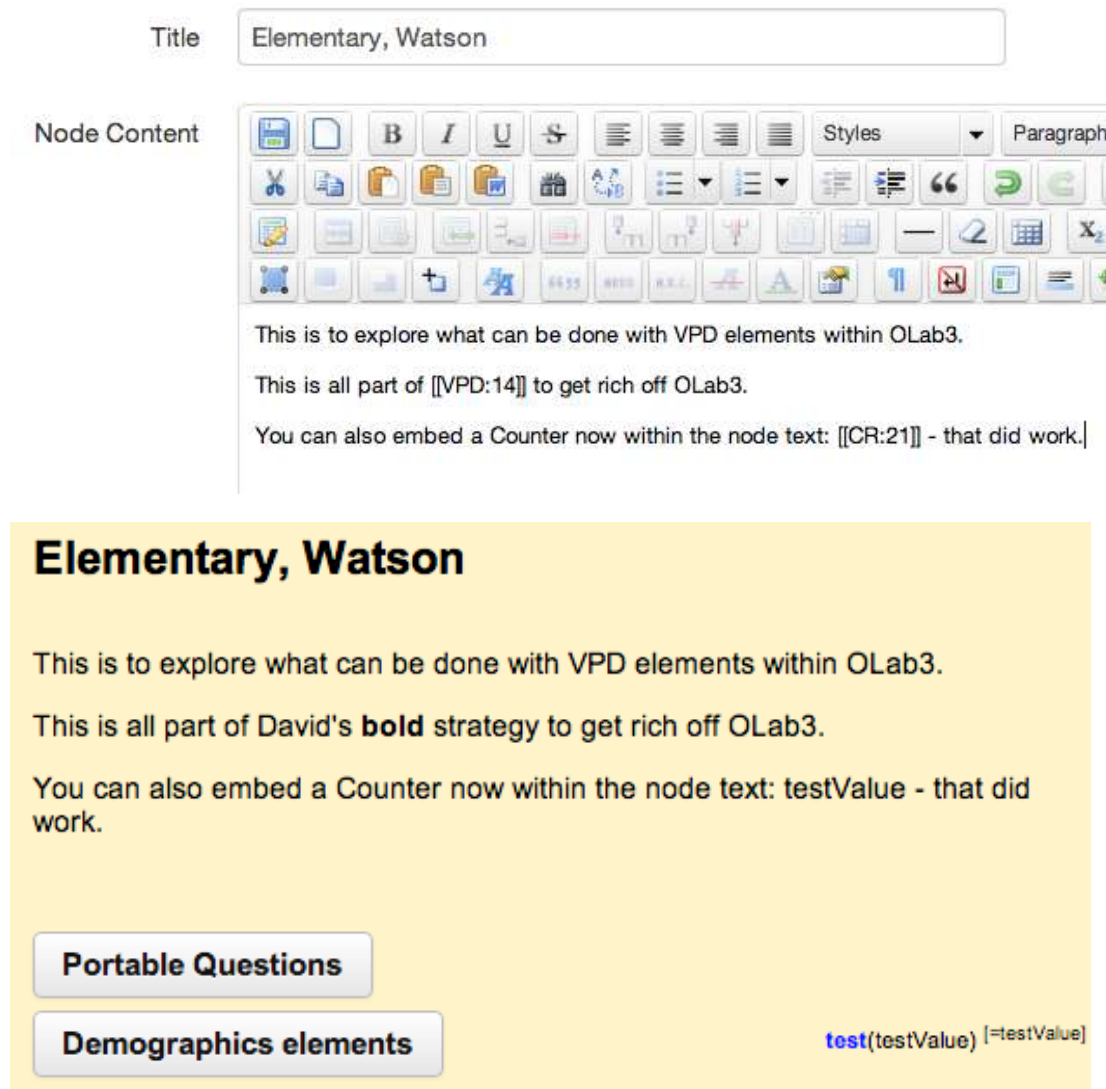


Figure 34 an embedded data element in the node editor (left) and how it renders on screen (right)

6.19 Clusters

Integrating elements – there are two ways of linking in VPD or MR elements to an activity node: as individually embedded elements or as a collection of elements via a data cluster.

Click 'Clusters' on the editor menu – this lists all of the current clusters. You can edit the contents of a cluster or delete a cluster or you can create a new cluster. Within a cluster you can add or remove data elements or media elements (such as pictures) and change the order in which they're shown to you.

Clusters have seen little use amongst OpenLabyrinth authors so far. They were introduced in OpenLabyrinth v2.5 to maintain compatibility with the ANSI/MVP specification. They will not be developed further at this point, unless we see a vigorous increase in interest.

6.20 Feedback

This lets you control how the feedback will be presented to the user at the end of the session. See [Feedback and Reporting](#). The Feedback Editor page itself is reasonably easy to follow. Various forms of feedback can be specified. There is a simple general text field where you can provide info that will be provided to all users.

You can add simple rules for the amount of time taken overall for the case. This is simpler and less flexible than the Timed Popup messages. You can add a simple rule which provides text feedback if a particular node was visited. You can also provide feedback if too many Must Avoid, or too few Must Visit nodes were visited. And lastly, you can provide feedback if a certain Counter value reaches a trigger criterion.

This is all much simpler and more limited than what can now be achieved with the more complex general [Rules](#) and largely dates back to OpenLabyrinth v2 functionality. However, it still works well and may suffice for some authors.

6.21 Skin

See section [Customization: Skins and Mashups](#)

6.22 Files

You may wish to add images, sounds, videos, documents, spreadsheets or other discrete files to your OpenLabyrinth map. To manage your files click on 'Files' link in the main menu. This launches the file manager window (see Figure 35: map files editor.).

The screenshot displays the OpenLabyrinth file manager. The top section shows a preview of a screenshot of a medical software interface. Below this, two files are listed in a table-like format:

File Name	Size	Last Modified	Actions
Poly Meds.png	202.278 KB	19.03.2013 21:05:19	edit, delete, image editor
FARMERCIE, POLLY 2013-01-18 TOXICOLOGY SCREEN	56.109 KB	20.03.2013 22:13:41	edit, delete, image editor

The 'FARMERCIE, POLLY' file is a text document containing the following content:

```

FARMERCIE, POLLY      2013-01-18 TOXICOLOGY SCREEN

AMPH      NEG      0.01
BARB      NEG      0.00
BENZ      POS      3.12
CANN      NEG      0.05
OPIOD     POS      1.13
COCA      NEG      0.00

CONFIRMED ACROSS TWO SAMPLES
SENSITIVITY & SPECIFICITY COMPARISON CALIB
INTERPRETATION OF THESE RESULTS SHOULD
BE MADE WHEN TRUE POS TEST POS
  
```

Figure 35: map files editor.

For each file uploaded into the current labyrinth you have:

- A wiki-style file reference: this takes the form of `[[MR:xxx]]` where MR indicates that this is a media resource and xxx is the unique file ID assigned to the file on upload. By pasting the reference into a node content box the file will be displayed at runtime.
- A resource preview: this just shows what the resource looks like (images only).
- A metadata view and editor: this shows basic metadata for the resource, allows you to edit the metadata and it allows you to delete the file from the current labyrinth.

In addition you can upload a new file by choosing it and clicking the upload button (see for list of supported file types'). Once a file has been uploaded it should be edited for its metadata (this will allow for both better control of the resource as well as registration with a learning object repository – see section on [Details](#)). Note that you should always ensure that any material you use (both text and images) is not covered by any copyright or consent restrictions. Liability for inappropriately used materials rests with the author.

File Type	Display	Extension	Notes
JPEG image	Inline	.jpg, .jpeg	JPEGS, GIFs and PNGs are the only supported image formats in OpenLabyrinth
GIF image	Inline	.gif	
PNG image	Inlinde	.png	Newly added in v3.1
Acrobat PDF	Link	.pdf	PDFs may be created from many applications including Adobe Acrobat
Shockwave Flash	Embed	.swf	Runtime media files from Adobe Flash.
Microsoft Word	Link	.doc, .docx	Standard document formats
Microsoft Excel	Link	.xls, .xlsx	
Microsoft PowerPoint	Link	.ppt, .pptx	
Rich Text Format	Link	.rtf	
QuickTime video	Embed	.mov	Video formats
MPEG-4 video	Embed	.mp4	
Windows Media	Embed	.wmv	
Real Stream (RAM)	Embed	.ram	
Real Stream (RPM)	Embed	.rpm	
Flash video	Embed	.flv	
MP3 audio	Embed	.mp3	Audio formats
WAV audio	Embed	.wav	

AAC (m4a) audio	Embed	.m4a
-----------------	-------	------

Table 2: file formats supported by OpenLabyrinth

We do suggest that, since Apple has decided to discontinue support for Flash media for iOS devices (iPhones, iPads etc) that authors look to other approaches such as HTML5.

6.23 Users

A map's authors are the only ones who can edit it and if the map's security is set to private they are also the only ones who can see it. By default the user who created a map (by any of the creation methods) is an author on the new map. Additional authors can be added (or removed) using the map's author editor which is accessed by clicking on the 'Tools | Manage Users & Groups' link in the main menu. This provides a list of all the registered OpenLabyrinth authors that can be selected to add the user as an author of the current map. Delete them to remove them from the current map. Note that you cannot add or remove yourself as an author.

Learners can also be added to a map so they can access it (but not edit it).

OpenLabyrinth supports LDAP authentication. Your system administrator can establish links from the OpenLDAP interface to your local LDAP server. This allows you to control larger groups through your normal access control lists and service management tools. See [Authentication systems](#) for more information about this.

6.24 Sessions

This lets you view all of the sessions run on the current labyrinth. See [Feedback and Reporting](#).

6.25 Export

There is now only one kind of export – MedBiquitous VP:

- You can export a labyrinth to the ANSI/MedBiquitous Virtual Patient standard package format – see section [Creating a OpenLabyrinth Map by Importing a MedBiquitous Virtual patient Package](#) for more details. To create an MVP package click on 'export MVP' from the editor main menu, select the appropriate licence to release the package and then create and download the zip file.

6.26 Duplicate

Any labyrinth you have edit access to can be duplicated as a template for a new labyrinth by clicking its 'duplicate' link on the editor page. A basic copy of the original labyrinth is created (called "Copy of ...") allowing you to change any aspect while keeping the original untouched. Note that duplicating just takes the basic structure and does not import files, rules or other additional properties.

For many purposes, a more effective way of duplicating a labyrinth is to Export it in MVP format, and then Import it again in MVP format. This has the additional benefit of making an archival copy on your local machine. It takes a few more steps to do this, but it also helps to retain the correct references to embedded Files, Questions and other OpenLabyrinth objects.

6.27 Author Notes

NOTE: This approach is being deprecated in OpenLabyrinth, from v3.1 onwards. Instead, we encourage authors to make use of the Notes fields to annotate items within a labyrinth, such as the overall labyrinth Details panel, as well as Nodes, Files, and other items to be included in future versions.

A simple note taker can be run for the authors of a labyrinth. Click on the pencil icon to load the author notes editor. This can be viewed and changed by any of the editors of any given labyrinth. This will be enhanced in future versions by semantic linking to external metadata tables, SPARQL endpoints and IMS-LTI consumers.



6.28 Key Feature Problems and Matching

NOTE: This approach is being deprecated in OpenLabyrinth, from v3.1 onwards. We encourage authors, as an alternative, to explore text-matching and [Rules](#).

If the labyrinth type is set to Key Feature Problem then any node within that labyrinth can be set to be a matching question rather than providing links. Turn matching on by going to any node in the current KFP typed labyrinth and scroll down to "add text matching", turn it on and submit the page. Go back and re-enter the node edit page and there is now an 'edit button' next to the "add text matching" buttons. Click edit to start adding different strings to be matched (up to 12 per question as well as the score for a match and whether the match is critical or not).

You can also set the following:

- counter controlled - this is the counter that is changed as a result of matching and adds the associated score to this counter
- strict [off][on] - this changes whether a string must be matched strictly or not - for instance if the match term is "cardiac arrest" if you enter "cardiac" they will get a match only if strict is turned off.
- number of options for user - this is the number of text boxes and therefore tries the user has to submit candidate matches
- next node - this is the node that you are taken to after the matching has been processed

In a matching-enabled node you are presented with a series of text boxes and a submit button - see figure 36.

edit matching

add the number of items to match, enter up to ten matching variations (case insensitive - blanks ignored) with scoring, the counter this will change and whether matching will be strict (exact match) or not

tension pneumothorax	- 1	- yes
hemothorax	- 1	- no
hypovolemic shock	- 1	- no
	- 0	- no
	- 0	- no
	- 0	- no
	- 0	- no

counter controlled Question 1

Question 1

An 18 year old male involved in an MVA arrives to the ER by ambulance. He is tachycardic with a heart rate of 136/mn. Decreased breath sounds on the left hemithorax, trachea is shifted to the right. SpO2 is 82% on room air. What are the likely causes of this patient's condition?

Submit

7. Feedback and Reporting

As with any educational program a labyrinth is at its most useful when it can inform a learner how well, or how badly, they performed. It is even more useful when accompanied by suggestions about how they might improve their performance in future. Feedback is an essential part of any educational activity. Although the content of each node is a form of feedback about the choices the learner made, OpenLabyrinth also supports summative feedback that is only provided once the session is complete.

7.1 Session Reports

As has already been mentioned each labyrinth user session is tracked and is available to its authors as a report. A user session is started each time you go to a map's root node. The tracking involves recording each node selection along with the time and current score and counter values at that point. The reports are available in one of two ways:

- Users can see a report for the most recent labyrinth they have run, if the author has provided this option. To do this first enable the node property of 'link to end and report from this node' for any node from which you want to allow the user to get their report – see [Nodes](#) for how to do this. The user simply needs to click on the link '**End session and view report**'.
- Authors can see reports for every session within a particular labyrinth – this is in the Sessions link on the left side of the main map editor.

In the Sessions Reports page, the author can see a dated list of all the occasions on which this particular labyrinth has been played, in the Sessions tab. Beside this tab at the top of the page, you will also see tabs for 'Path visualization' and 'Aggregate report'. More on these shortly.

If you click on the date/time link in the table of Sessions, you can examine what occurred on that occasion. The session report provides the username (unless played anonymously), when and for how long the labyrinth was played and a quick count of nodes visited. Even this brief information can be very useful in comparing what a user got out of a session.

- Metadata such as the user ID, the session ID and the labyrinth name.
- The start time and the time taken to complete
- The total number of nodes visited as well as the number of 'must visit' and 'must avoid' nodes visited
- General feedback irrespective of what you have done
- Feedback in response to specific nodes visited
- Feedback in response to the number of 'must visit' and 'must avoid' nodes visited
- Feedback based on the time taken to complete
- Feedback: on the final values of any of the current labyrinth's counters
- The list of nodes visited
- The histogram of time spent per node

- The graph of counter values through the session

This includes any specific feedback generated by [Feedback](#) items specified by the case author. For some cases, this might be quite extensive, but is often absent altogether if nothing was specified by the author. The next section shows the specific responses to Questions. (This still needs a bit of fine tuning.)

There follows a table, and histogram of time spent on each node- (see Figure 37). This is more often useful for debugging cases to see where users get stuck or bored. Finally, there is a simple graph showing how Counter values changed during the session – see figure 38. In simple cases, this is easy to follow but in large cases, the node numbers on the x-axis start to get pretty messy.

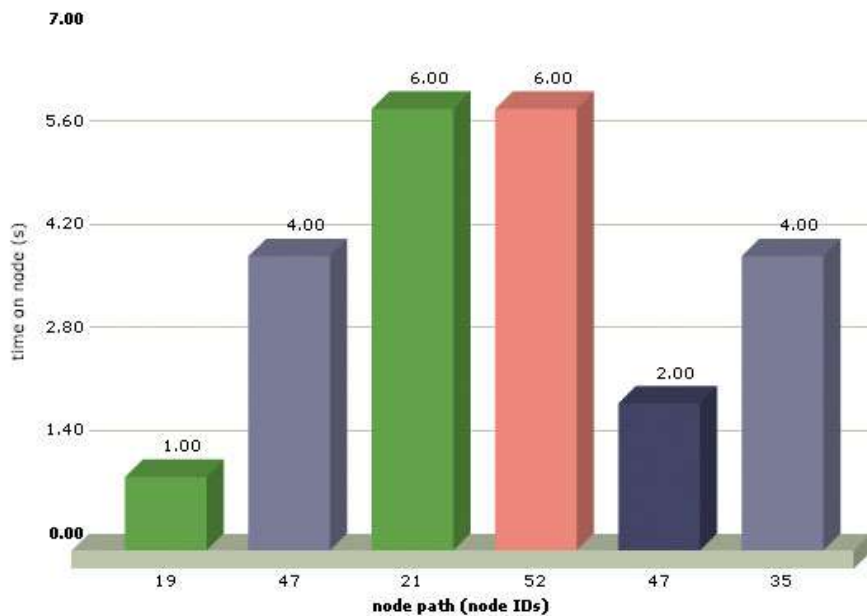


Figure 37: a typical OpenLabyrinth report histogram.

In this session you spent 6 seconds on nodes 21 and 52. Node 21 is a 'must visit' node (in green) and 52 is a 'must avoid node' (in red).

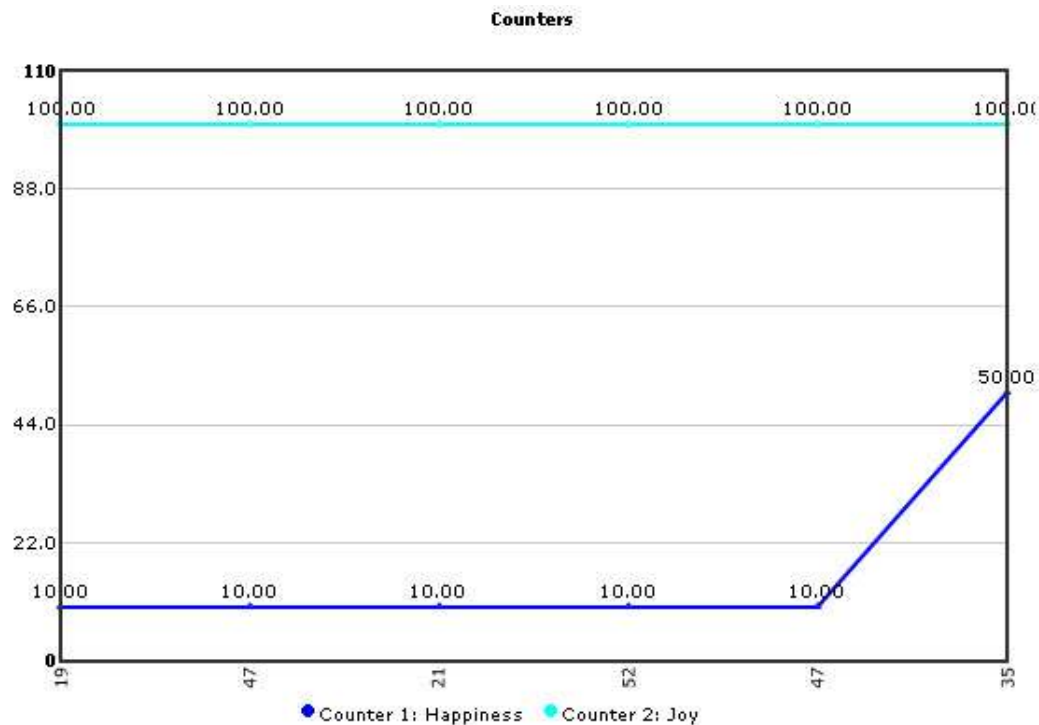
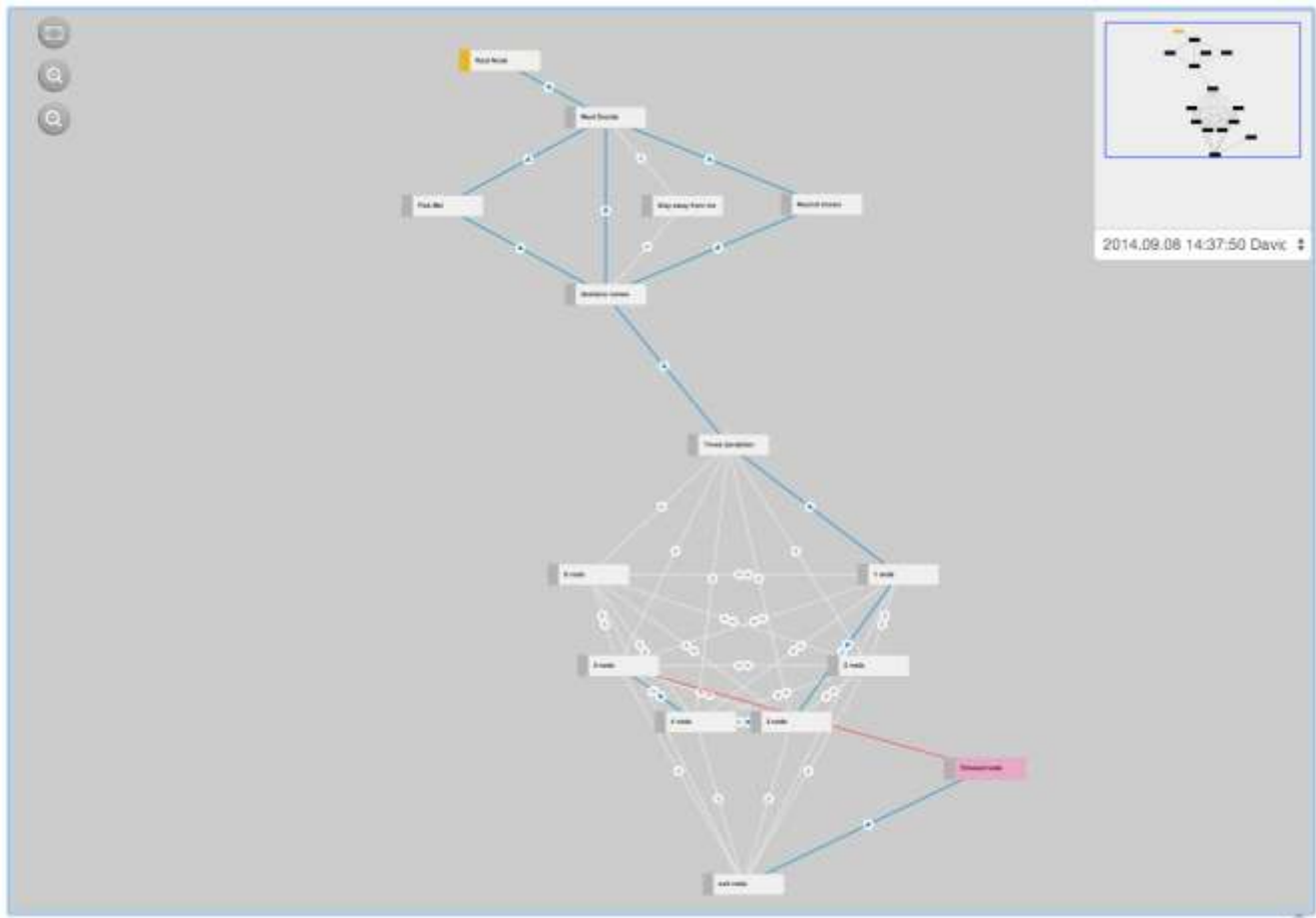


Figure 38: a typical counter trace

There are two counters here 'happiness' which started at 10 and rose to 50 on the last node and 'joy' which remained at 100 throughout

The data can be exported as a XLS file for further analysis in Excel or similar tools.

In the Session Reports master page, there is a tab for 'Path visualization' – this is proving to be a surprisingly useful tool for some projects. This report uses the same visualization tool as the Visual Editor, and has the nodes laid out in the same pattern.



When you pick the session that you want to examine from the drop-down list in the top right corner, the node map will be highlighted with the paths taken by the user. A red path indicates an jump enforced by a Rule. This path visualization can be particularly useful for getting a sense of whether a user just blasted through a case, without exploring possible options, or possibly wandered in circles quite lost – perhaps you did not make the options clear enough? Some good feedback for the case authors.

The third tab on the Session Reports page provides aggregate data for all the sessions recorded so far.

7.2 Feedback Options

Each labyrinth can be set to provide feedback on user performance on the following areas:

- General feedback to you – for instance “That was a hard problem ...”
- Feedback in response to specific nodes visited – for instance “You chose to get an MRI, that was an expensive option given the circumstances ...”
- Feedback in response to the number of ‘must visit’ and ‘must avoid’ nodes visited – for instance “You made 4 choices that should have been avoided ...”
- Feedback based on the time taken to complete – for instance “You completed the task in less than 2 minutes, were you really thinking about what you were doing?”
- Feedback: on the final values of any of the counters – for instance “Your morale slipped below the critical point ...”

User feedback can be configured using the '[Feedback](#)' option in the editor menu.

8. Rapid Reporting of Real-time Responses (4R)

There is now another option for providing rapid feedback to Users. This is tied in with Scenarios and is particularly useful for synchronous group work.

8.1 Creating a 4R report

For our Dynia project, we needed a rapid way of providing feedback to participants so that they could compare choices and answers. For this, we developed 'Rapid Reporting of Real-time Responses (4R)'.

This is at present, for project purposes, tightly integrated with the OpenLabyrinth [Scenario Manager](#) but this capability may be made available as part of general feedback in OpenLabyrinth eventually.

First, you need to create a Scenario, and have some users play the case before you can see a 4R report. Follow the steps in [Scenario Manager](#) for how to do this.

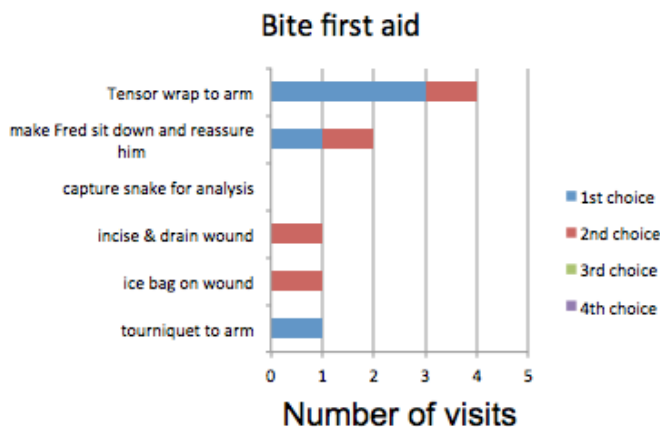
However, you also need to indicate which nodes are to be included in the 4R report. It works best when comparing groups of nodes that were presented as a set of choices, as in a dandelion.

The 4R report is a snapshot, for collaborative group use, of certain Sections within a labyrinth. It is not intended to replace the more extensive Session Report generated when a user clicks on the 'End Session and View Feedback' link at the end of a case.

8.2 Which choices in a Dandelion?

As a reminder, a dandelion is the name that we give to a group of nodes in a labyrinth where each node is accessible from the other. They are called this because of the puffball appearance in [Visual Editor](#) for a large group.

One simple way to use a 4R report is to look at which choices were most popular, and in which order were they chosen.



In the 4R graph above, the users had a number of first aid measures for a snake bite. Looking at this graph 'tensor wrap to arm' is the most popular choice and most users made it their first choice, and one user made it

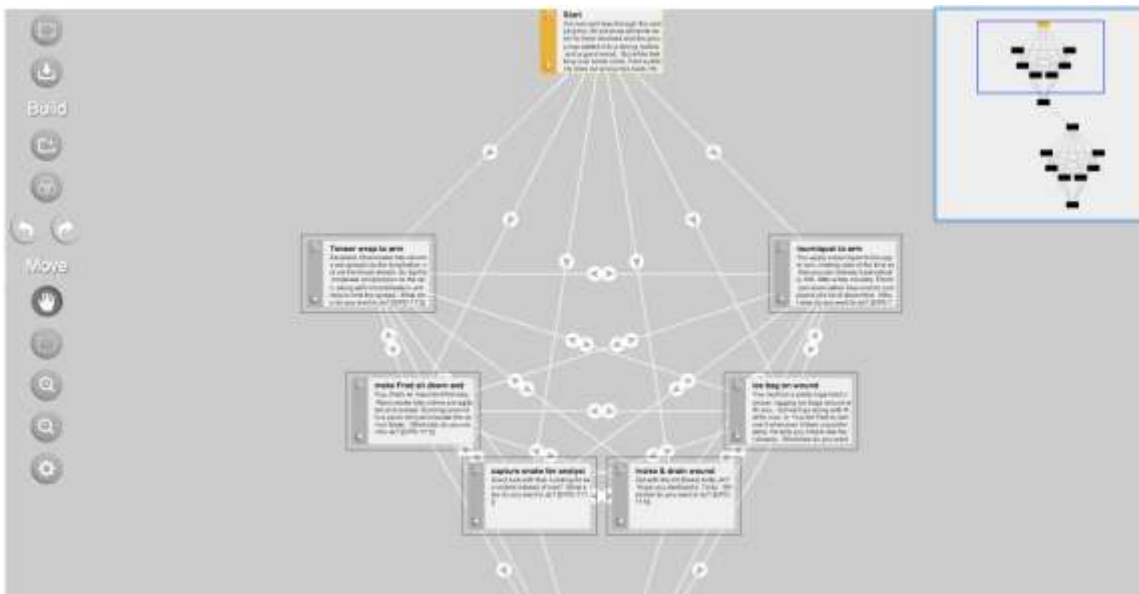
his second choice. Two other nodes were picked as first choice ('make Fred sit down' and 'tourniquet to arm') and three nodes were picked as second choice ('make Fred sit', 'incise wound' and 'ice bag'). Nobody caught the snake and nobody made more than two choices. However, OpenLabyrinth would go on to record up to 4 choices made by the users in the Scenario.

Note that the 4R report preserves user anonymity to some degree and does not show which user made which choice. More detailed analysis can be gleaned from the underlying database but is not part of the 4R.

('Number of visits' is an odd term that has been changed to Frequency. It simply means how many users visited or clicked on this node.)

To indicate which nodes are to be included in a 4R, we use [Sections](#). Sections were previously used in OpenLabyrinth only for case navigation. They now have greater utility and can be used to group nodes together for various purposes.

To select Nodes into a Section, the easiest way is to use the Visual Editor.

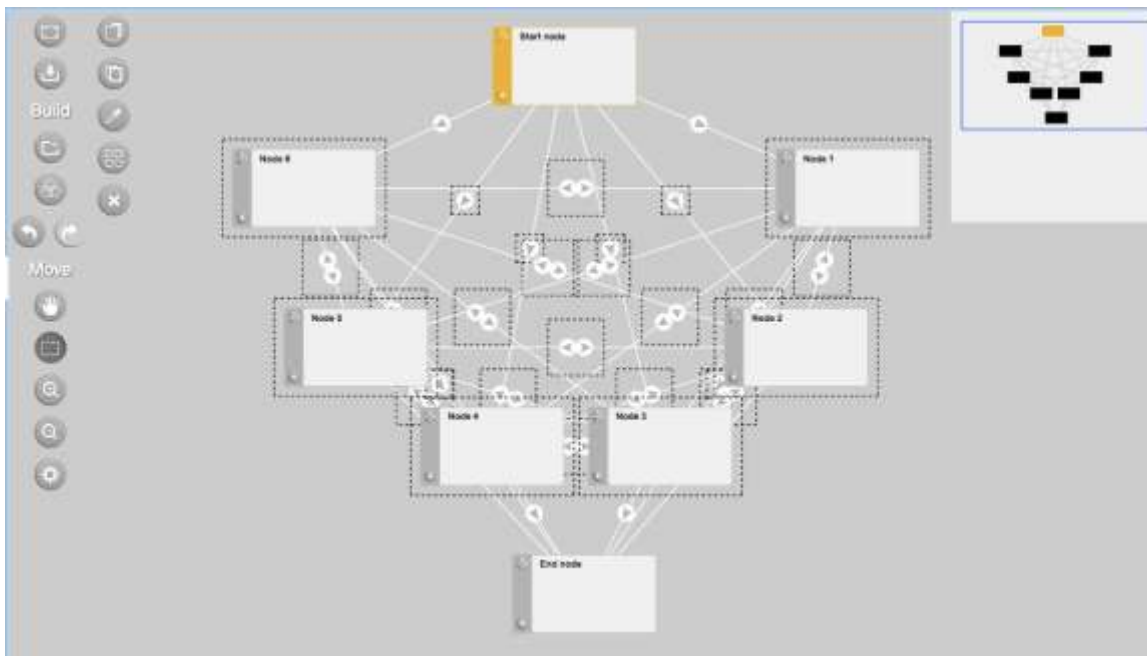


It is a little hard to see on this image but each of the nodes in this dandelion have a secondary colored outline. We have created a YouTube screencast, www.youtube.com/watch?v=dYepQfp_BLY, to illustrate this in more detail. Nodes with the same color of secondary outline are all in the same Section. (This screencast also addresses how to edit 'Prevent revisit' nodes in a similar manner.)

At present, for 4R purposes, a node should not be in more than one Section, in the Visual Editor. There is no cross-checking for this. Case authors will need to be careful about this when selecting nodes into Sections.

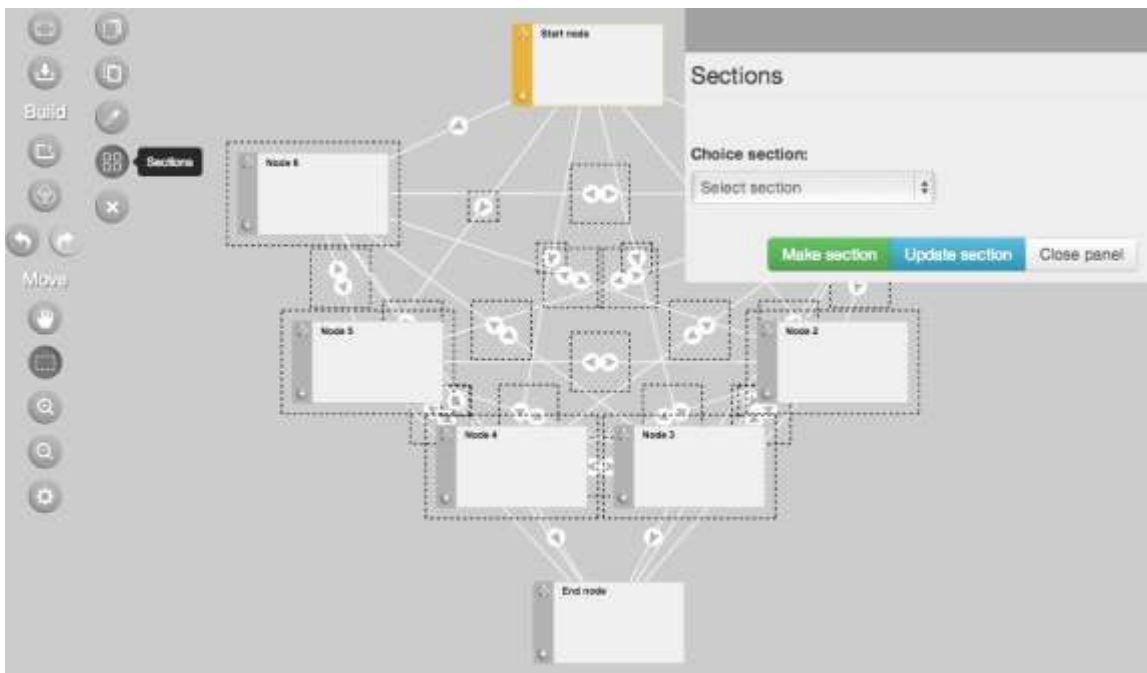
8.3 Creating a Section

In the Visual Editor, use the Selection tool to highlight the Nodes you want with a draggable rectangle. All of the selected Nodes will then be shown with a dotted outline.



You will now see a new set of tools in the top left corner of the Visual Editor. These actions can be applied to the selected set of Nodes. From the top down, these icons are Copy, Paste, Change color, Sections and Delete selected.

Click on the Sections tool (with 4 small squares in the icon).



The Sections dialog box will appear. You can make a new Section with the green button or you can pick from existing Sections in the drop-down list.

Click on 'Make section', you will then be able to name it and also alter the order in which Sections appear in a 4R report.

New Section

Section name:

Node 1

0

Node 2

0

Node 3

0

Node 4

0

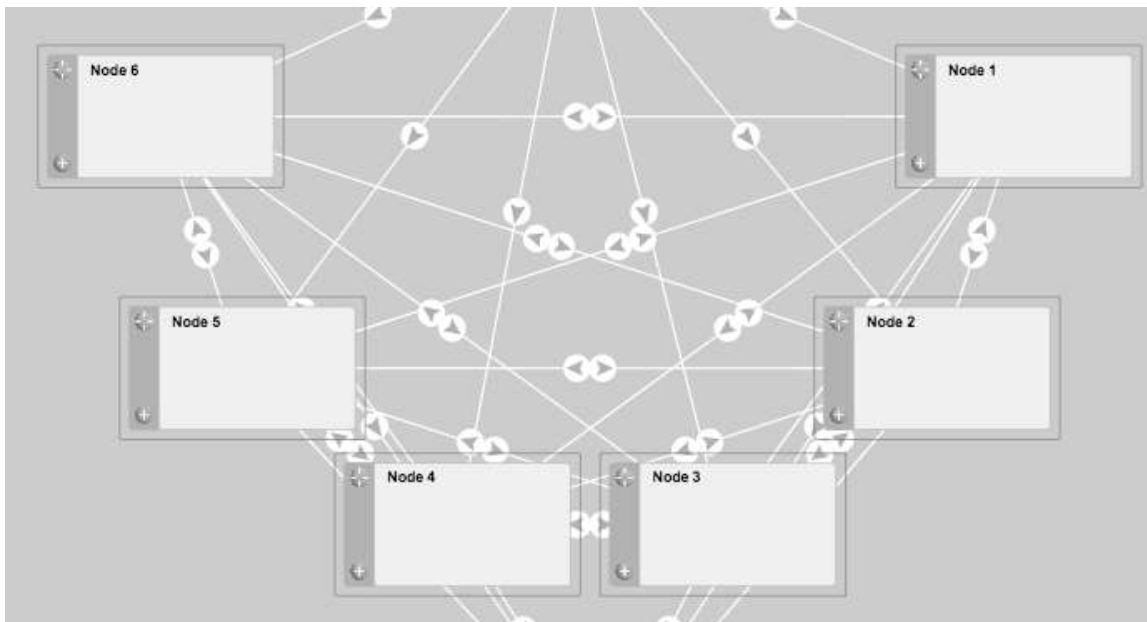
Node 5

0

Save Close

For illustration of how the node order changes in the 4R report, I have kept Node 1 first but reversed the order of the remaining nodes.

Note that the nodes in that dandelion have a subtle secondary outline. If there is more than one Section in a labyrinth, each Section will have a different color.



Remember to save your changes by clicking the Update button in the top left of Visual Editor before you leave.

8.4 Manual edit of a Section

You can manually edit which nodes are in a Section and in which order they are reported by using the Sections editor – see [Sections](#) in left sidebar menu of OpenLabyrinth.

Section title	Nodes	Operations
Test section	"Node 1" - ID:8999 - order:0 "Node 6" - ID:9004 - order:1 "Node 5" - ID:9003 - order:2 "Node 4" - ID:9002 - order:3 "Node 3" - ID:9001 - order:4 "Node 2" - ID:9000 - order:5	Edit delete
Test section	"Node 1" - ID:8999 - order:0 "Node 6" - ID:9004 - order:1 "Node 5" - ID:9003 - order:2 "Node 4" - ID:9002 - order:3 "Node 3" - ID:9001 - order:4 "Node 2" - ID:9000 - order:5	Edit delete

Add a node section

Title

[Add section](#)

Close scrutiny of the above list will show you a wee bug in the Visual Editor creation of Sections. Duplicate sections are sometimes created. For now, it is easy to delete extra Sections using this manual editor.

One other wee quirk in the order number to the Nodes: it uses base zero for counting. This makes sense to programmers but some users might find that a bit odd. A future fix?

8.5 Generating a 4R Report

One key point, which arises from how we have integrated the 4R report with the Scenario Manager, and which will eventually be simplified, is that the Scenario Manager needs to know when the User has completed a labyrinth.

End node

[End Session and View Feedback](#)

At present, we make use of the 'End Session and View Feedback' link. (This is a temporary workaround for now.) The User must click this link to indicate to the Scenario Manager that he has finished the labyrinth.

Note that the labyrinth author must enable this link. Usually this is done in the last Node of the labyrinth. In the Node editor, the last option for the node...



Link to end and report from this node ☐ Off (default) ☒ On


...must be turned on.


Now, when the Scenario director who is running the Scenario sees that all the Users have completed the Step, he can click on the small blue eyeball icon to view the 4R report.

Scenario Progress

Users

Step A  

Sections and 4R analysis 

D Topps (gmail) 

☒ Include in report

✓

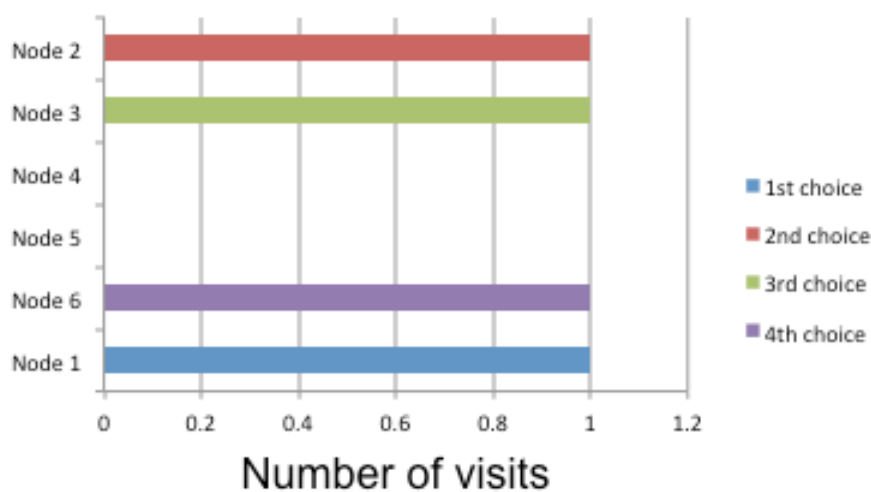
[Go Forum](#)

The 4R report for this test case looks like this...

Sections and 4R analysis

Node ID	Count base	1st choice	2nd choice	3rd choice	4th choice	1st choice %	Title
8999	6	1	0	0	0	16.67%	Node 1
9004		0	0	0	1	0.00%	Node 6
9003		0	0	0	0	0.00%	Node 5
9002		0	0	0	0	0.00%	Node 4
9001		0	0	1	0	0.00%	Node 3
9000		0	1	0	0	0.00%	Node 2
9005		0	0	0	0	0.00%	Start node
9006		0	0	0	0	0.00%	End node

Test section



In this particular report, I was lazy and only had one user. Looking at the above, there are a few things of note. The table shows you that this User chose Node 1 first, then Node 2, Node 3 and then Node 6.

The graph is slightly easier to interpret, with blue being the first choice etc. The graphical output makes more sense when you examine the choices for multiple Users, as in the first graph in this document.

Look at the order of the nodes names in the vertical axis of the graph, you will notice something else, when you compare to the Node order as specified in the Section. The Node names are listed in order, but go up from the graph origin in the bottom left corner. This is mathematically correct but is slightly counterintuitive to non-math users, who expect to read from the top down. We will probably reverse the order in which nodes are listed so that the first node appears at the top of the vertical axis, furthest away from the graph origin.

8.6 Looking at more than choice order

For this document, we have focused on order of choices made within a dandelion.

The 4R report is more powerful than that. If you include QQuestions within the Nodes you have selected in your Sections, such as pick-choice, multiple choice or slider QQuestions, then the responses of Users is also displayed in the 4R report.

[generate some better reports which illustrate how these work]

9. Scenarios

9.1 Scenario Manager

Based on the principles described in Ruth Colvin Clark's book, 'Scenario Based e-Learning' (ISBN-13: 978-1118127254), we have introduced the concepts of Scenarios into OpenLabyrinth.

Scenarios hold multiple labyrinths together as a group. For example, we can use Scenarios to group topics such as low back pain or headache but Scenarios are different from [Collections](#). Both are sets or groups of labyrinths. A Collection is a simple set that a superuser has grouped together which may have a feature in common e.g. all labyrinths associated with a project, or a set of how-to labyrinths.

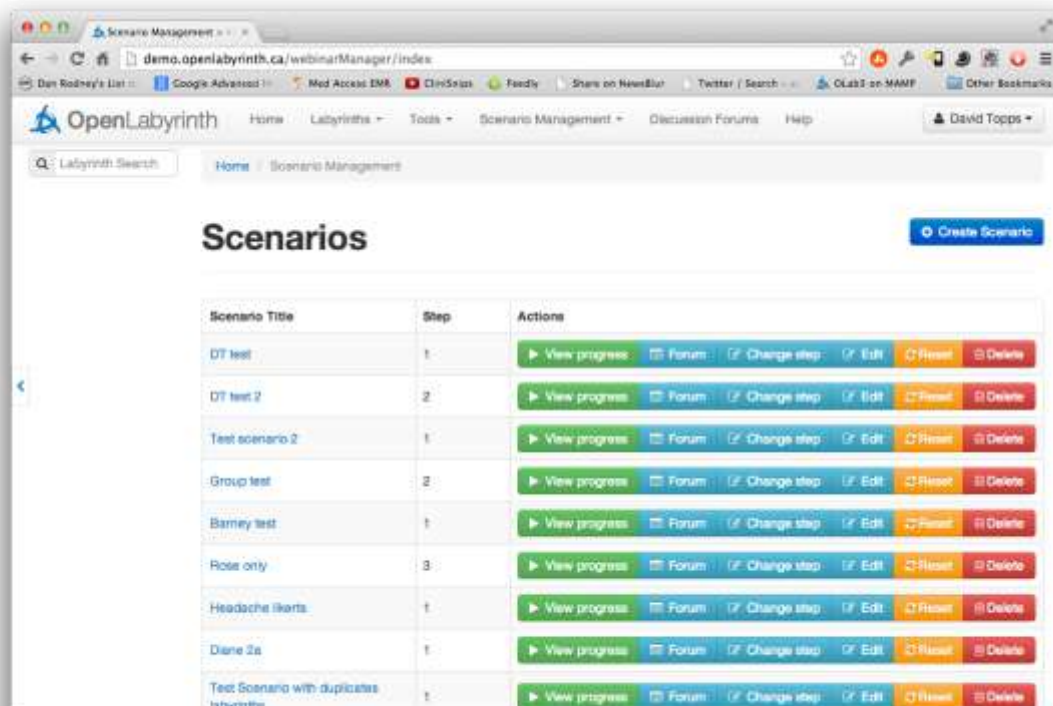
A Scenario is much more powerful, providing control over such things as the order in which the labyrinths are played, when they are accessible, and who has access to them. We can also use Scenario Manager to connect to the relevant discussion [Forum](#), and provide the link to the [4R report](#) for Learner feedback.

At present, a Scenario is still a bit limited and currently represents a sequence of labyrinths, presented to a Group of Users, along with an integrated discussion Forum. In future versions of OpenLabyrinth, we will be expanding this capability to allow integration of other learning activities, allow for longitudinal or parallel sets of linked labyrinths, and provide some control over which user can progress from one labyrinth to another.

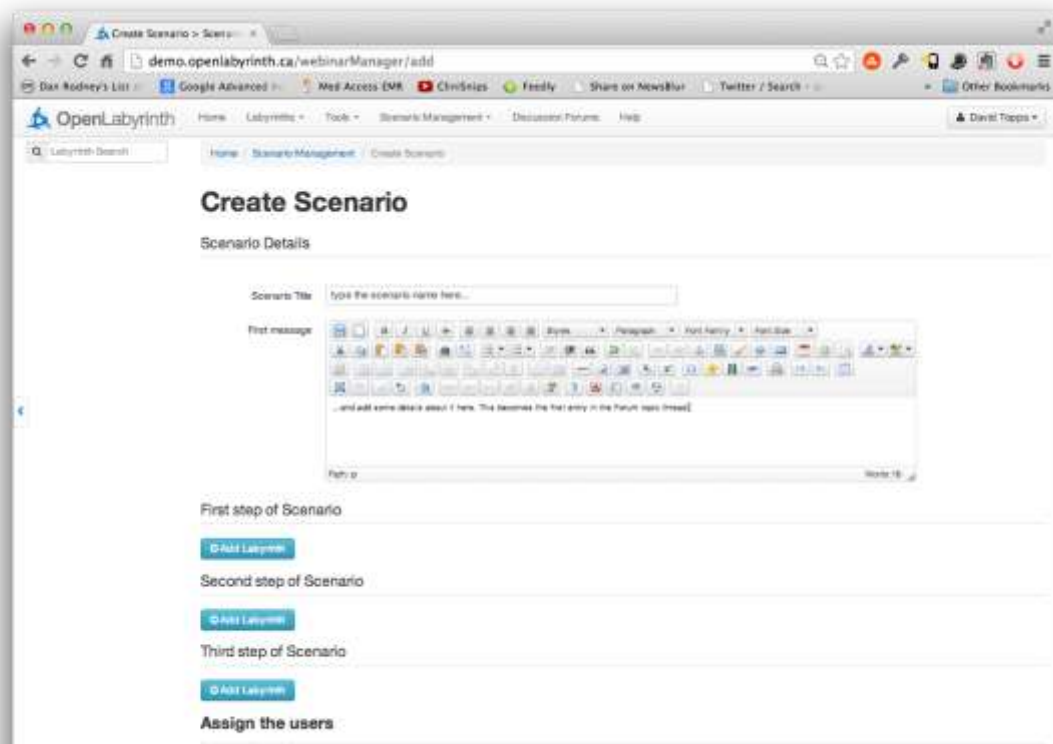
Scenarios will also take over the function of Presentations (which are soon to be deprecated).

9.2 Creating a Scenario

To create a Scenario, click on the top drop-down menu 'Scenario Management' and select 'Manage Scenarios'. You will see the following screen...

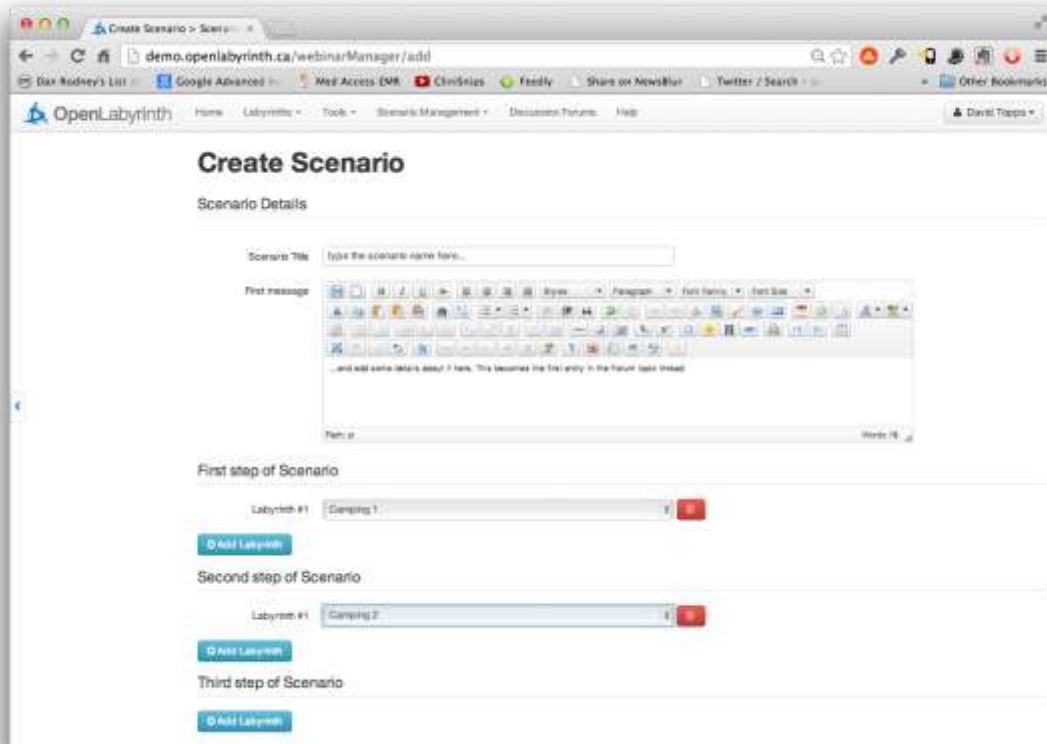


Next click on the 'Create Scenario' button in the top right corner. On the next screen...



...enter the name of the Scenario, plus details about it. The details become the first post in the related Discussion Forum topic. Next click on the blue button 'Add labyrinth' for the first step. Choose from the list of available labyrinths.

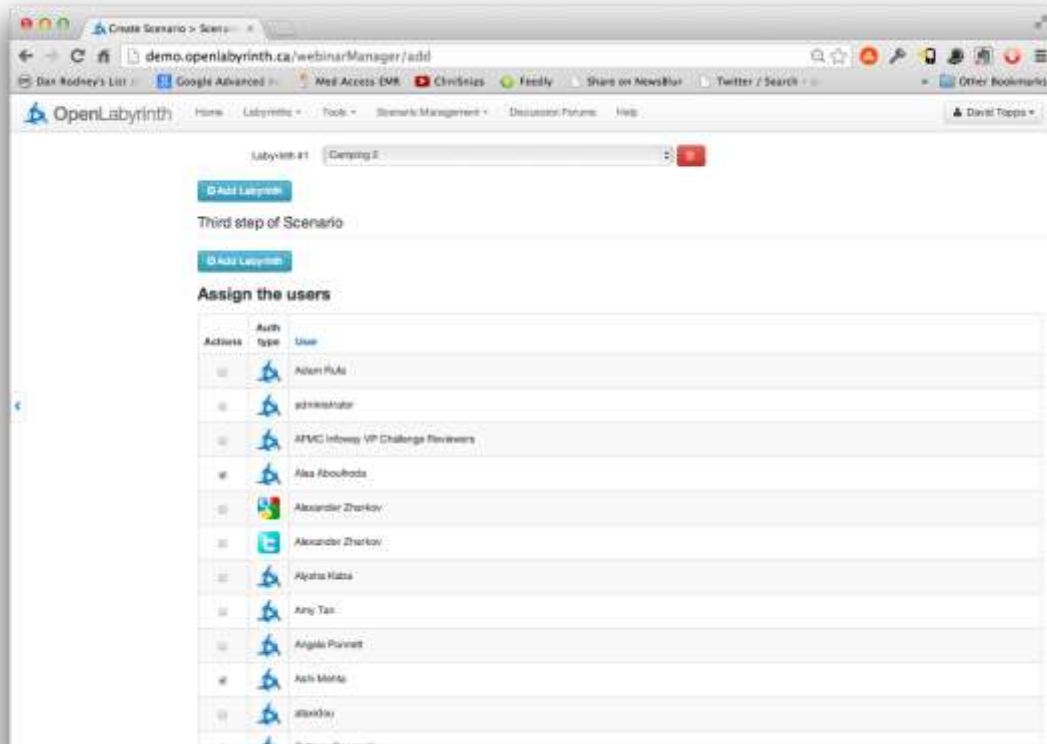
Do the same for the second step of the Scenario, in the same manner. Alternatively, for some Scenarios, you can add more than one labyrinth into the first Step.



The screenshot shows a web browser window with the URL `demo.openlabyrinth.ca/webinarManager/add`. The page title is "Create Scenario". Under "Scenario Details", there is a text input for "Scenario Title" and a rich text editor for "First message". Below this, the "First step of Scenario" section contains a dropdown menu for "Labyrinth #1" with "Camping 1" selected, and a blue "Add Labyrinth" button. The "Second step of Scenario" section has a similar dropdown for "Labyrinth #1" with "Camping 2" selected and another "Add Labyrinth" button. The "Third step of Scenario" section is partially visible with a third "Add Labyrinth" button.

If needed, you can also add one or more cases to the third Step. You can have as many cases per Step as you want.

Next you must assign some Users to the Scenario. You can either check off names from the list or you can assign a Group of Users.



Finally, click on the blue 'Create Scenario' button down in the bottom right corner to save your Scenario.

If you need to make changes to an existing Scenario, simply click on the blue Edit button beside the Scenario's name in the overall Scenario Manager.

9.3 Using Scenarios

To start using the Scenario, click on the green [View progress] button beside the Scenario name. You will see a display of your Steps and assigned Users in the Scenario Progress screen...

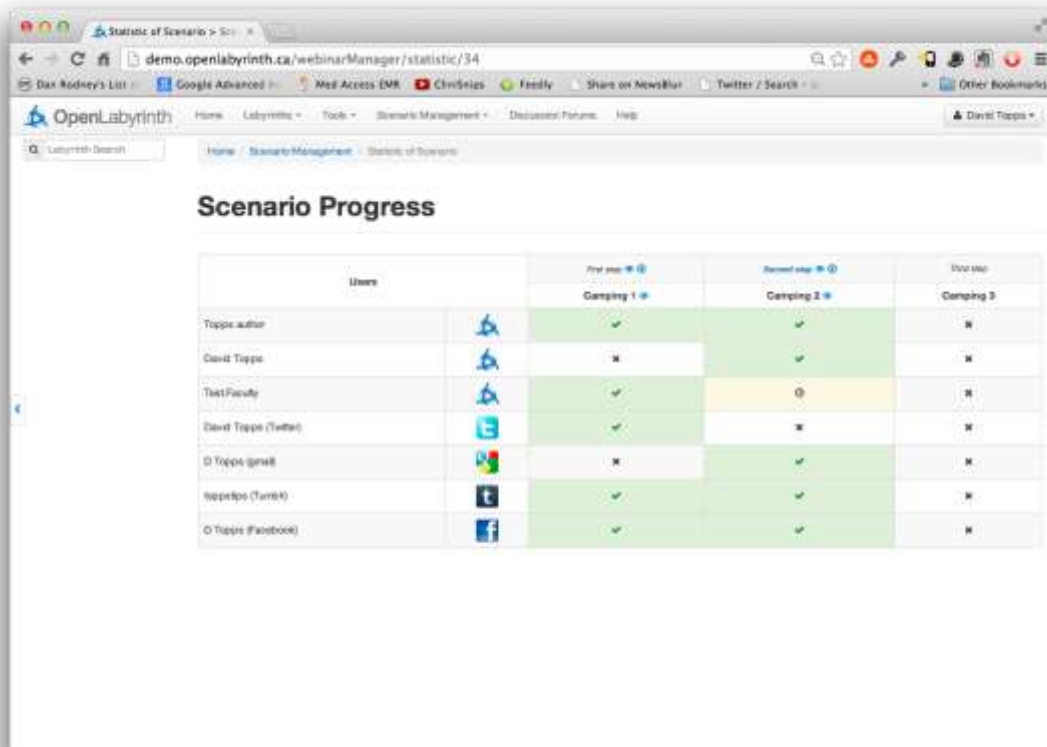
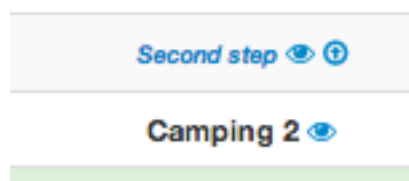


Figure 39: Scenario Progress view

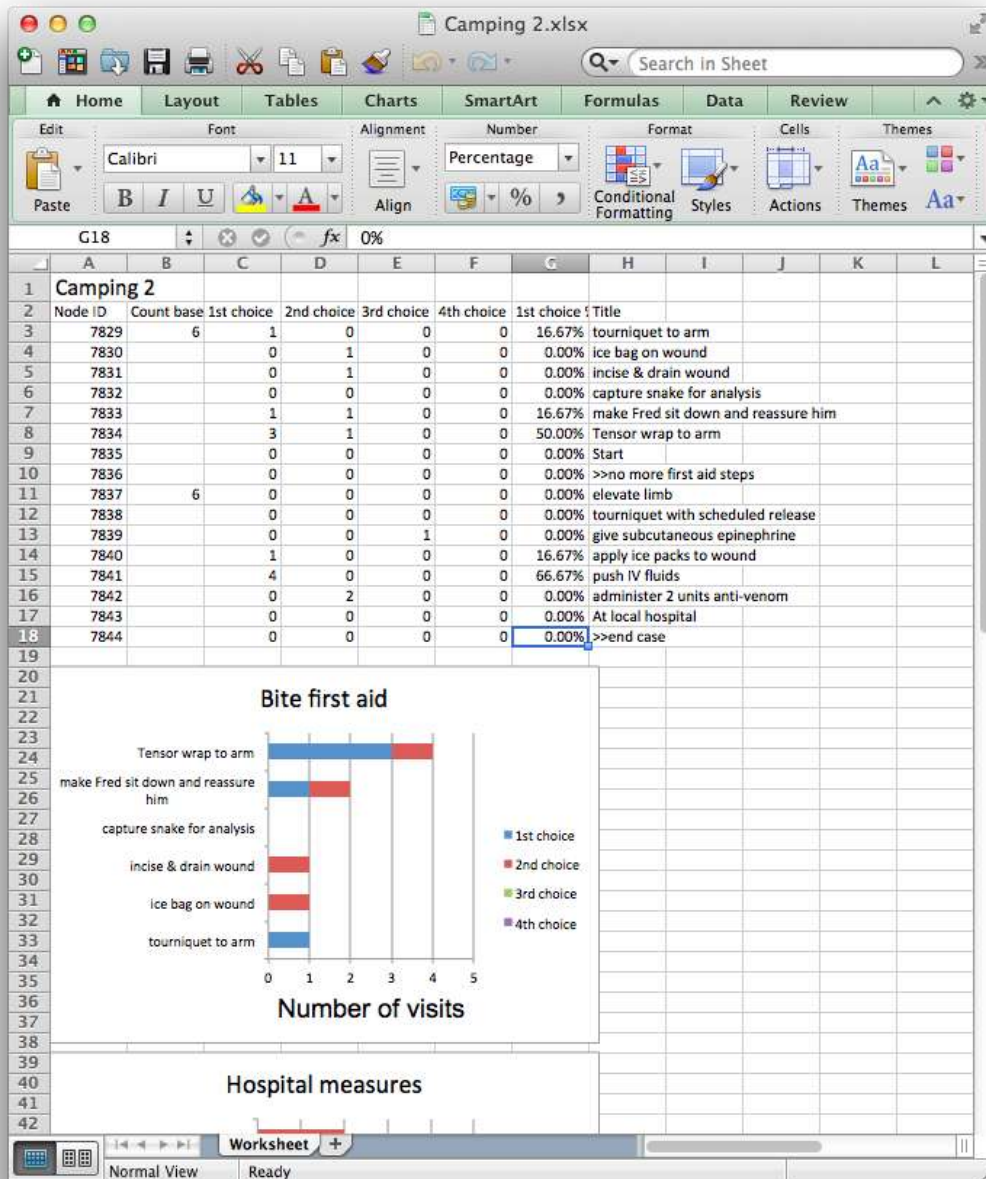
This screen shows a Scenario part way through. In the above screen, five out of seven Users have completed the first Step. The second Step is now running – five users have completed the case, one is still running the case (yellow bar) and one has not started.

Look carefully at the above screen, you will see some small blue icons beside the labyrinth names...



Click on the small icon that looks like an eye, OpenLabyrinth will generate a 4R report for that case. It appears in your browser's download list as a Microsoft Excel spreadsheet, with the same name as the labyrinth. The other small blue icon with small up arrow will send a copy of the same 4R report spreadsheet to the Discussion Forum.

If you open the Excel spreadsheet, you will see a worksheet like the following...



First you will see a table of results, which are accurate but hard to make sense of. Scroll further down the worksheet; you will see the horizontal bar charts, which are easier for the participants to interpret.

Each bar chart represents a Section in the labyrinth. We have used Sections in the same manner as for basic labyrinth navigation. (For information on how to allocate Sections with the Visual Editor, see [Creating a Section](#) under the chapter on 4R reports.) Basically, you select a group of nodes using the box selector, then click on the Sections button. Add a new Section and name it to something that makes sense to you.

Now the other side to this is what does the participating User see? How do they launch one of the labyrinths in the Scenario? When the User logs in, they should click on the top Scenarios menu and then My Scenarios...

My Scenario

Scenario Title	Step	Actions
Camping mid size group	2	<div>Open</div>

Click on the green [Play] button beside the relevant Scenario...

Scenario 'Camping mid size group'

First step

#1 Camping 1

Second step

#1 Camping 2
View 4R

Third step

#1 Camping 3

Go to Discussion Forum thread

In the above window, this user has already completed the second Step case for this Scenario. Otherwise there would be a green [Play] button beside it, which launches the labyrinth in the usual style.

At this point, this user simply navigates that labyrinth case in the usual manner. It works best if they remember to exit the case at the end by clicking the ‘End Session and View Report’ link – this tells the Scenario Manager that that User has finished and that cell goes green in the Scenario Progress table.

At the moment, the Report that the User sees with ‘End Session and View Report’ is the generic style (and not very useful) Session report that is a holdover from OpenLabyrinth version 2. Soon, we will have more intelligent linking into the full Session Reporting systems and the 4R reports.

In the above window, the User can also view their personal 4R report by clicking on the blue ‘View 4R’ button. This is not available until they have completed the case in that Step.

9.4 Scenario-based Learning Design

Scenario based learning (SBL) is a powerful overarching concept that covers many of the activities and designs that we already use in simulation and virtual patients. The underlying architecture of OpenLabyrinth is very favorable for creating a more extensive SBL design tool. If you are interested in exploring these areas of educational research with us, please contact us. We may also create an area on our web site with example SBL designs, if there is interest in this.

There are already very powerful Learning Design tools out there. For example, LAMS (<http://www.lamsinternational.com>) is a very powerful tool and has become highly developed. However, we have found it to be too powerful and broadly focused, which makes it less useful for medical education. As we expand the SBL components of OpenLabyrinth, we will be taking a more focused approach.

10. Forums

OpenLabyrinth has its own internal Discussion Forum embedded into it. This is very simple in function. As an alternative, you can link via the IMS-LTI interface (<http://www.imsglobal.org/toolsinteroperability2.cfm>) to external software that provides such functionality, such as Moodle or Desire2Learn.

A Forum is generally attached to a labyrinth or to a Scenario. When linked to a Scenario, it allows the Group of Users enrolled in that Scenario to mutually discuss items of interest from the case. Each time an item is posted to the Forum, its participants are notified by email so that they don't have to continually check the Forum for new responses. This can be turned off for each User.

Creating a new Forum is simple enough that we won't go into detail here, on the assumption that most admins who are setting up Forums will be very familiar with the basics from other areas of activity. Users can be individually assigned to a Forum, or you can also assign Groups of Users.

When you receive an email notification of a Forum post, there is an attached link that will take you directly to that point in the Forum.

Forums

[Add new forum](#)

Forum name	Users	Comments	Last
Forum	Count of users	Count of comments	Last comment
Feedback testing David Toppe 2013-12-23 11:51:34	1 users	1 comments	David Toppe 2013-12-23 11:51:34

Within the listing of Forums that are open to you, you can simple click on the Forum title to explore its contents. If you are the Forum owner, you can also add Topics, Delete the Forum, or Edit things like the participant list, forum details etc.

- ITrex Test Topic
 - second topic
 - First test Topic
- Message:

B I U L ↺ ↻ Styles Paragraph Font Family Font Size

Path: p Words:0

Add message

Because the Forums have proven to be such an effective tool for internal use during case authoring, we will also be integrating them into the upcoming [collaborative authoring](#) functions in OpenLabyrinth.

In addition to the creation and running of OpenLabyrinth maps there are a number of OpenLabyrinth administration services including user management, reporting and import/export:

All authors need to have a login to OpenLabyrinth. OpenLabyrinth super users have the ability to create OpenLabyrinth accounts using the link from the home page. Each account requires a username, password, full name and email address. You can also set options such as the security level (learner, author, reviewer, superuser); user interface language; and simple or advanced interface level. You can choose to automatically notify the User via email of their login details. Users can reset their passwords, or they can be reset by a superuser.

Edit user account

username

password

name

email

superuser ☒

language [english ☒

Figure 40: the add user screen

The various security access levels are used to determine the level of access for a particular user and what they can do within OpenLabyrinth. Note that this also depends to some extent on what security level of the labyrinth has been set to.

A = all access; E = edit; P = Play; Pk = P only via key; S = play via Scenario

	open	closed	private	key
superuser	A, P	A, P	A, P	A, Pk
author	E, P	E, P	E, P	E, Pk
reviewer	P	P, S	P, S	Pk
learner	P	P, S	P, S	Pk

See [Permissions](#) for more details on case security. To explain the above table a bit further, a superuser can do pretty much anything on that server, including access to all cases, user accounts, reports etc. An author can do anything to their cases, including grant access to other authors. Reviewers and Learners are progressively restricted on what they can do.

[Scenarios](#) are one other approach to consider in controlling access to when a labyrinth can be played. This is particularly useful for Groups, or when you want a set of labyrinths to be played in a particular order.

Groups are sets of users. The tools for creating groups and managing their membership are below those for users on the same page. If you need to provide a more complex method of administering your users and groups, we suggest that you connect your OpenLabyrinth server to an LDAP server, or to an IMS-LTI Consumer. See [Authentication Systems](#) for more information on this.

Note that a User can only have one level of access. If you want to provide a User with author level access to some cases, and reviewer level access to others for example, you will need to create two different user accounts for them.

11.2 Language support

The user interface language is set in the user account creation or editing pages. If you change your own user language then you must logout and log back in for the changes to take effect. Currently there are just two supported languages – English and French. Note that the user guide is currently available only in English although a French translation would be most welcome. Note that the user interface phrases are stored in the “interface.xml” file sitting in the documents folder at the OpenLabyrinth root.

At present the French translation is likely to be pretty poor limited as it is to the author's memories of school French and the quality of Google Translate. Corrections and improvements are invited. Furthermore anyone willing to translate the 370 or so interface elements into a third or fourth language are welcomed and encouraged. Simply add a language tag at the top of the XML file – eg <language ID="KN" name="klingson" /> - and then add an additional element for each phrase – eg <phraseKN>qapla</phraseKN>. Return to the author for inclusion in the next release.

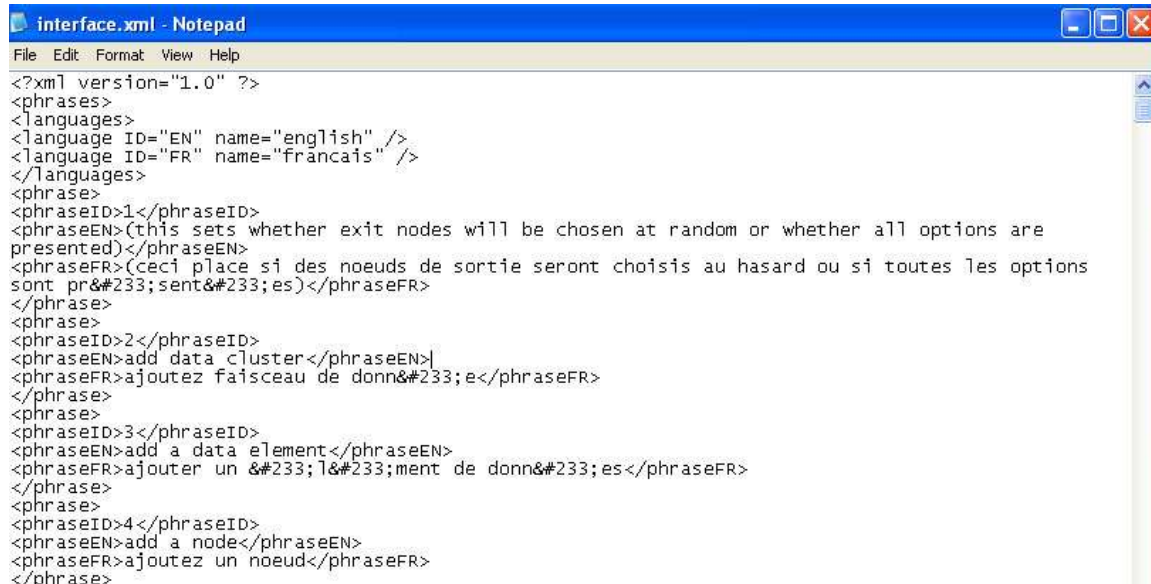


Figure 41: the interface.xml file that can be found in documents/interface.xml

11.3 Presentations

NOTE: Presentations will be deprecated sometime after v3.11 – their functionality will be incorporated into [Scenarios](#). Authors who want to create a particular look and feel in their labyrinths are encouraged to explore the use of [Skins](#), which are much more powerful than Presentations.

Groups of Labyrinths can be collected together as 'presentations' for instance as a course page or a bank of assessment items. To create a presentation click on the 'presentations' link on the home page, add the details for the presentation such as any text to be shown and user options and then submit. Edit a presentation to add specific labyrinths to it. Note that you must be an author on any labyrinth you want to include in a presentation – see figure 42.

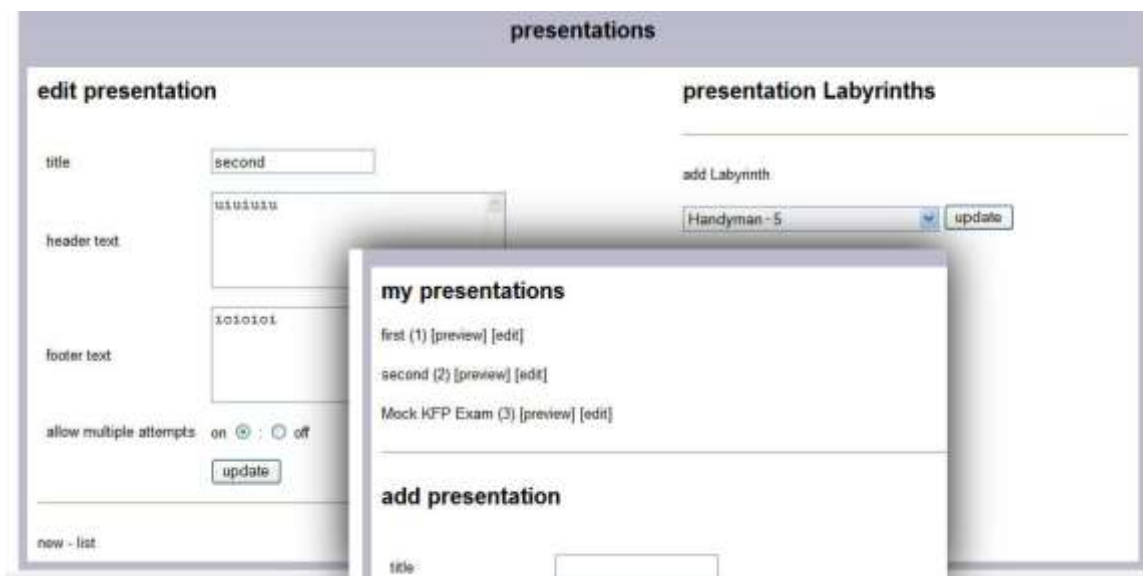


Figure 42: the Presentations authoring screens

11.4 Collections

This allows you to create a collection of different labyrinths. For instance, there may be a set on a particular topic or that use a particular approach. A labyrinth can be allocated to multiple collections.

11.5 MedBiquitous Virtual Patient Export

OpenLabyrinth currently exports to the MedBiquitous Virtual Patient XML format. To create a package, click the menu item Labyrinths | Export MedBiquitous. This will run a script that creates a new folder, copies the required files into it, writes XML files for the various required components and then creates a zipped archive of the folder and its contents – the user is then presented with a link to the zip archive.

The key elements are labelled as follows:

- A The Zipped package
- B The IMS Manifest XML file that lists the contents of the archive
- C The metadata XML file in LOM format
- D Activity Model XML file
- E Folder containing all of the media objects
- F Virtual Patient Data XML file
- G DAM XML file

The other files and folders are required objects for a SCORM package.

11.6 MedBiquitous Virtual Patient Import

In a similar but reverse way to MVP export, OpenLabyrinth can create a new labyrinth by importing and MVP package. To create a labyrinth, click the menu item Labyrinths | Import MedBiquitous. This will run a script to upload an MVP package.

You select the zip archive, this is uploaded to a new temporary directory, unzipped, the XML files parsed and the data written into the database, files copied to new directories and when all the importing has been done the temporary directory deleted and a link to the new labyrinth presented to the user.

This works best with MVP packages created by OpenLabyrinth v2.6.1 or later. Prior versions miss out quite important detail in their packages.

11.7 Migrating cases from OpenLabyrinth version 2 to version 3

Both versions of OpenLabyrinth are highly compliant with the ANSI/MedBiquitous Virtual Patient standard and this is the easiest way to migrate cases from one version of OpenLabyrinth to another. Simply Export your labyrinth as an MVP Export from the OpenLabyrinth2 server and then Import to the OpenLabyrinth3 server. Depending on how the original labyrinth was crafted, this may be a very simple process but there are certain things to be aware of.

In the last version of OpenLabyrinth v2.6.1, the MVP Export module was a bit incomplete. Because OpenLabyrinth 2.6.1 is no longer being updated, there is no set mechanism at present for fixing this. We have created a modified version of the `mvp_export.asp` module that more completely exports all the data needed. If you are having trouble importing MVP zip files created by your OpenLabyrinth2 server, we suggest that you ask your system administrator to copy our version of `mvp_export.asp` over the existing one. You will not see a difference when running the Export process but the zip file that is created will contain better data.

Here is a current list of OpenLabyrinth2 features that may not survive migration across servers:

- a. Hard coded links to local resources – this will seem obvious to many but not all clinical authors understand the difference between relative and absolute references.
- b. Direct links to URLs on external web sites – these should be maintained. But we have found sometimes that the link is clobbered. It is a good idea to keep your original cases running so that you can compare before and after and copy such links over.
- c. Ordered Links – these usually become random again. To be fixed soon.
- d. Avatars sometimes look blank at first – if you open the Avatar editor, make a minor change and then save it, the original should appear.
- e. Skins are not automatically transferred along with cases. Originally Skins were designed be a way for sites to apply their own look and feel to a case and so remain specific to a server. It is possible but not simple to copy Skins from one OpenLabyrinth server to another.
- f. Users – because user authorisation is managed locally on a server, we cannot migrate user permissions along with a case.

The majority of what is important about a case will be successfully migrated – all the node text, embedded images and other Media Resource files (those found in the Files section of a case), the links and pathways, the Questions, Counters and metadata should all be there. Please let us know if you find problems.

11.8 Migrating cases between OpenLabyrinth version 3 servers

The most reliable way is also to use the MVP Export and Import process as described above. Some of the same caveats apply about things that do not survive migration, with the same fixes. From v3.2 onwards, we now offer 'Advanced Export' of labyrinths. As the capabilities of OpenLabyrinth have expanded, we can no longer accommodate all the functionality within the Medbiquitous MVP spec. Advanced Export can be used for more sophisticated cases and includes all current functionality. Note that the two OpenLabyrinth servers will need to offer equivalent functionality in their versions.

Note also that currently OpenLabyrinth3 is still in a state of rapid evolution. If the two servers are not in roughly the same state with regards to having installed the latest code from Github, then some features or underlying tables may have changed. It is unlikely that you will lose data through this, but you may have to do some additional editing to restore a customized feature.

11.9 Importing MVP formatted cases from other software

The ANSI/MedBiquitous Virtual Patient standard is your best bet for moving cases between different software applications. While the MVP standard is well defined, it has not anticipated all of the directions in which our very active virtual patient authoring communities have taken their software.

There are a few things to consider when migrating cases across software.

- a. Linear cases are generally easier to migrate than branched cases. All of the offspring of the original Labyrinth, developed in Edinburgh, support branching cases. Many other VP authoring platforms do not support branching or only do in a very limited manner.
- b. The cases are data driven, which means that the underlying SQL data tables contain most of the essentials. SQL is a common database platform, supported on many operating systems. This means that it is not important as to what OS the server is installed upon... in most cases. MS Windows can be quirky as always, which is why we have moved to a LAMP setup (Linux, Apache, MySQL, PHP).
- c. Because the user runs the cases on a web browser, there is generally less dependence on which browser is used. No software needs to be installed on the user's machine to run a case. But not all web browsers are well behaved. For example, many VP platforms are very dependent on Adobe Flash, including OpenLabyrinth2. Apple has refused to allow Flash on their iOS devices. For this reason, we have abandoned Flash in OpenLabyrinth3 and are only using HTML5, the new current standard.
- d. But not all web browsers properly support HTML5 – you can't win! This is a known problem. Notably, MS Internet Explorer 6, 7 and 8 do not support HTML5. There is nothing we can do about this. We suggest that you use a modern web browser such as Chrome or Firefox.
- e. We have been successful in importing cases into OpenLabyrinth3 from OpenLabyrinth2, the original Labyrinth (Edinburgh), DecisionSim, vpSim. But usually a bit of tweaking is required.

12. OpenLabyrinth Remote Services

[NOTE: the OpenLabyrinth remote web services have seen very little use – because of this, we have not implemented these in OpenLabyrinth v3. The text in this section has been copied from the OpenLabyrinth v2 manual mostly as a placeholder for now. If there is a strong demand for web services, we may resume this. Or we encourage other groups to consider implementing this functionality, in the spirit of open-source development.]

In addition to users running a labyrinth within OpenLabyrinth itself, there is a web-service that allows activities to be run remotely. To run a remote OpenLabyrinth service you need to have a remote service (including a single client IP address) registered within OpenLabyrinth, one or more OpenLabyrinth maps associated with that service, and a means of presenting the activity to the user in the remote environment. This OpenLabyrinth 'client' application would need to be able to consume the XML generated by the service and render it so that the user was given the links and services required to run the labyrinth remotely and interact with the OpenLabyrinth server.

12.1 Description

In addition to playing OpenLabyrinth activities within OpenLabyrinth itself, you can also run them in a remote system such as a virtual learning environment or e-assessment system by using OpenLabyrinth Remote

Services. These services use XML to send OpenLabyrinth node content to the remote client that then communicates back using simple URLs (see figure 43).

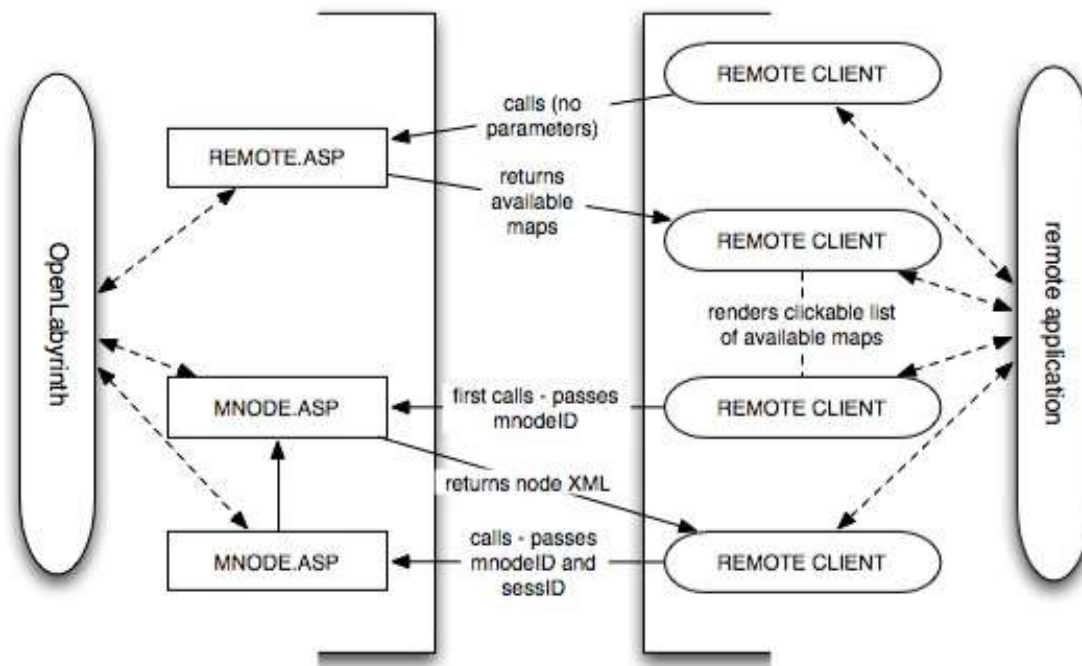


Figure 43: OpenLabyrinth Remote Services Architecture.

OpenLabyrinth is on the left and the remote application is on the right hand side. The steps are:

1. The remote client calls REMOTE.ASP in OpenLabyrinth
2. If the user credentials are correct or the remote client's IP is recognised then REMOTE.ASP sends back an XML file listing the available Labyrinth Maps
3. The remote client parses the incoming XML into clickable links to start each Labyrinth Map
4. When the user clicks on a link this sends a call to MNODE.ASP passing just the nodeId
5. If the call is authorised then MNODE.ASP returns the node XML package
6. The remote client parses the incoming XML so the user can work through the activity. Each click cycles through steps 4-6
7. As this is happening additional communication can run within each application

12.2 Setting up OpenLabyrinth Remote Services

There are two ways a remote service can be called:

1. by passing user credentials in the remote GET string. The user needs to be a remote type user (i.e. not a superuser, author or learner account). To set this up:
 - create a remote user or reuse an existing one
 - add the remote user to any maps you want to be available in the remote service
 - call the remote service by adding hashed userID and password credentials (uid=xxx&pwd=yyy) using the cookieHash function in UTILITIES.ASP . There is a commented function in REMOTE.ASP that allows you to generate the hashed credentials

2. by registering a remote service. To set this up:

- click on 'remote services' on the home page and then 'add a service'
- give the service a name and enter the IP address or range of the remote server. Leave box 4 or 3 and 4 blank to indicate a range of IP addresses, e.g. put 123 in box 1 and 45 in box 2 to accept any IP starting with 123.45.
- once the service has been created then add Labyrinth Maps to the service by clicking 'add/edit Labyrinths' and adding them from the dropdown menu

12.3 Remote Services Components and Messaging

There are at least three discrete functions required at the client end:

1. Client list: this queries OpenLabyrinth for the OpenLabyrinth activities assigned to the current remote service and then formats and displays the results of the query
2. Client player: this interacts with the MNODE.ASP script in OpenLabyrinth to run the activity
3. Client remote services: these support additional OpenLabyrinth services like info boxes, counters and DAM nodes

In addition there may be local services layered on top of the remote OpenLabyrinth application. The components and their accompanying messaging flows are laid out in figure 44.

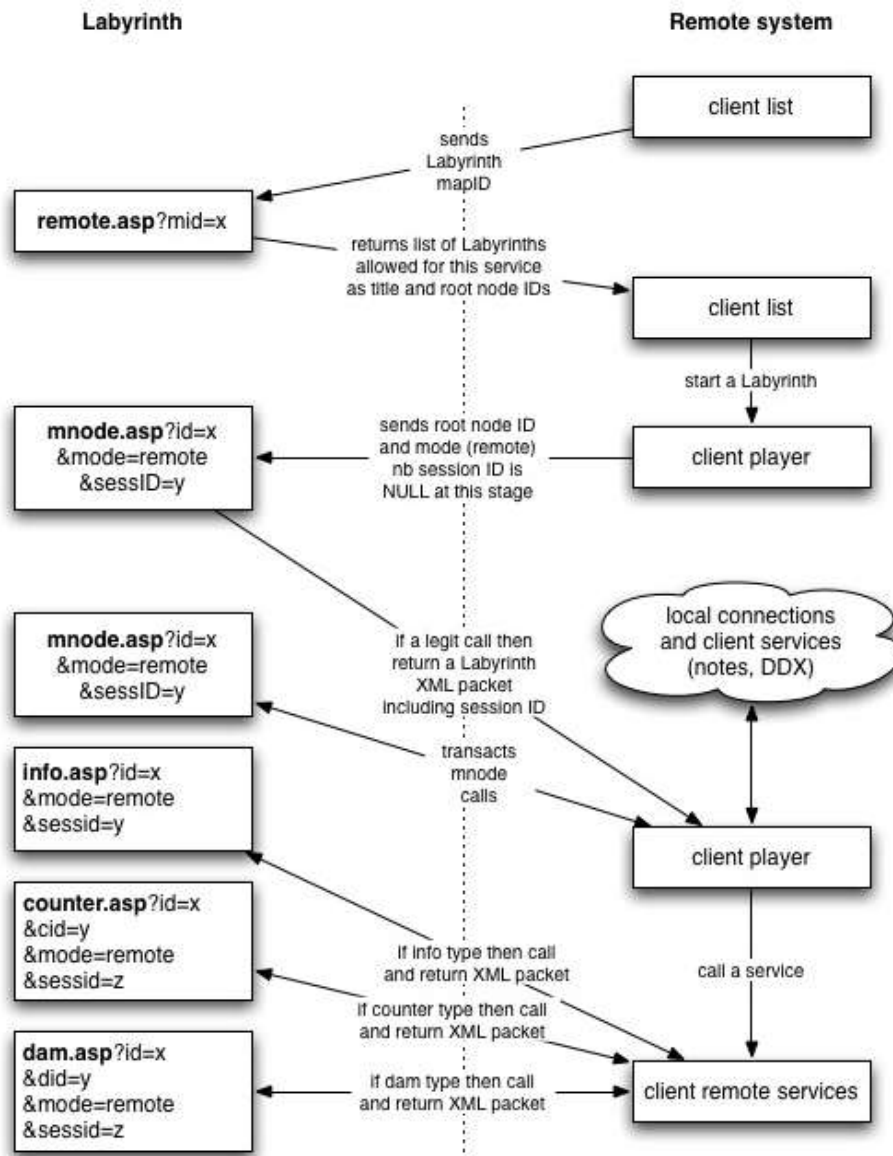


Figure 44: remote components and their interactions with OpenLabyrinth

12.4 OpenLabyrinth Remote Services Transactions

There are three transaction types in the remote services: OpenLabyrinthservice, OpenLabyrinth and service.

Transaction: OpenLabyrinthservice

This transaction is called by the client list function to get a list of the OpenLabyrinth activities available for the current remote service. The URL for the call is: <http://OpenLabyrinth.mvm.ed.ac.uk/remote.asp>

- `<OpenLabyrinthservice>` is the root element
- `<remoteIP>` is the IP address as seen by OpenLabyrinth
- `<OpenLabyrinthmap>` is the parent element per activity
- `<OpenLabyrinthmapid>` is the current activity's map ID (integer)

- <OpenLabyrinthmapname> is the current activity's title
- <OpenLabyrinthmaproot> is the root mnode ID (integer) for the current activity

As an example:

```
<?xml version="1.0"?>
<OpenLabyrinthservice>
  <remoteIP>129.215.133.15</remoteIP><OpenLabyrinthmap>
    <OpenLabyrinthmapid>174</OpenLabyrinthmapid>
    <OpenLabyrinthmapname>ESSQ+1</OpenLabyrinthmapname>
    <OpenLabyrinthmaproot>2054</OpenLabyrinthmaproot>
  </OpenLabyrinthmap>
  <OpenLabyrinthmap>
    <OpenLabyrinthmapid>178</OpenLabyrinthmapid>
    <OpenLabyrinthmapname>ESSQ+Jaundice+1</OpenLabyrinthmapname>
    <OpenLabyrinthmaproot>2129</OpenLabyrinthmaproot>
  </OpenLabyrinthmap>
</OpenLabyrinthservice>
```

Transaction: OpenLabyrinth

This is the main transaction type with OpenLabyrinth which processes the activity as it plays out.

The URL for this is: <http://OpenLabyrinth.mvm.ed.ac.uk/mnode.asp?id=x&mode=remote&sessID=y>

The value for sessID can be left blank for the root node transaction but is required thereafter to maintain scores etc. As long as the node ID requested is in an activity that is allowed in the current service an XML response is sent back with the following elements:

- <OpenLabyrinth> is the root element
- <mnodeitle> is the current node's URL-encoded title
- <mapname> is the current activity's URL-encoded title
- <mapid> is the current activity's (integer)
- <mnodeid> is the current node's ID (integer)
- <mapscore> is the current activity score (integer)
- <message> is the current node's URL-encoded body text
- <colourbar> is the current node's URL-encoded text indicating the use of colour bars for zoned scores (in game mode only)
- <linker> is the current node's URL-encoded HTML links
- <tracestring> is the current node's URL-encoded HTML for the trace/track
- <rootnode> is the node ID for the current activity's root node (integer)
- <infolink> is the current node's 'info' button URL-encoded HTML
- <usermode> is a toggle value between basic and expert
- <dam> is a list of any associated MVP Data Availability Model nodes
- <mysession> is the GUID for this particular user session
- <counterstring> is URL-encoded text of links for all the counters in this map
- <timestring> and <javascripttime> are deprecated in the current version - disregard

As an example:

```
<?xml version="1.0"?>
<OpenLabyrinth>
  <mnodetitle>Start</mnodetitle>
  <javascripttime></javascripttime>
  <mapname>ESSQ+Jaundice+1</mapname>
  <mapid>178</mapid>
  <mnodeid>2129</mnodeid>
  <mapscore>100</mapscore>
  <timestamp></timestamp>
  <message>%3Cp%3Eyou+are+a+FY2+in+general+surgery+on+call++you+are+phoned+by+a+GP+who+
has+a+55yo+woman++in+his+surgery+who+appears+jaundiced++what+do+you+do+now%3A%3C%2Fp%
3E</message>
  <colourbar>%3Ctable+border%3D%270%27+width%3D%27100%25%27+cellpadding%3D%275%27%3E%3C
tr%3E%3Ctd+height%</colourbar>
  <linker>%3Cp%3E%3DABBCE08B%2D1F75%2D4A0A%2D852B%2D21F0819A19A2%27%3EID%3D%5B27%5D%5D%
5D%5D+2D+arrange+urgent+review++the+patient+comes+to+hospit%3C%2Fa%3E%3C%2Fp%3E</lin
ker>
  <tracestring>%3Ca+href%3D%22%23%22+onclick%3D%22toggle%5Fvisibility%28%27track%27%29%
3B%22%3E%3Cp+class%3D%27style2%27%3E%3Cstrong%3EReview+your+pathway%3C%</tracestring>
  <rootnode>2129</rootnode>
  <infolink></infolink>
  <usermode></usermode>
  <dam></dam>
  <mysession>ABBCE08B-1F75-4A0A-852B-21F0819A19A2</mysession>
  <counterstring></counterstring>
</OpenLabyrinth>
```

Transaction: service

This is a combined function that connects to three different OpenLabyrinth services:

- Information: <http://OpenLabyrinth.mvm.ed.ac.uk/info.asp?id=x&mode=remote&sessid=y>
- Counters: <http://OpenLabyrinth.mvm.ed.ac.uk/counter.asp?id=x&cid=y&mode=remote&sessid=z>
- DAM Nodes: <http://OpenLabyrinth.mvm.ed.ac.uk/counter.asp?id=x&did=y&mode=remote&sessid=z>

Each of these scripts bundles their response as a single XML element:

- <OpenLabyrinth> the root element
- <service> a URL-encoded HTML text response

The following info button call is an example:

```
<?xml version="1.0" encoding="UTF-8"?>
<OpenLabyrinth>
  <service>Your+role+in+this+clinic+is+to+review+patients+on+behalf+of+your+Senior%2E+H
e+is+present+in+the+clinic%2C+and+will+call+in++to+see+if+you+are+progressing+well+in
+the+case%2E+++This+is+the+Virtual+Hospital+Environment+%28VHE%29%2C+take+advantage+o
f+the+fact+there+is+no+time+pressure+and+make+sure+you+understand+the+basic+sciences+
you+are+applying%2E+++Better+than+getting+a+textbook+out+in+front+of+the+patient%2E</
service>
</OpenLabyrinth>
```

12.5 Basic OpenLabyrinth Client Functions

The OpenLabyrinth client is relatively thin in that it only needs to send out appropriate and well-formed requests and to parse, format and present the returned XML data. As any free-text information is exchanged as hexadecimal data a converter from hex to plain text is required as is some data cleaning, for instance:

- Changing instances of "mnode.asp" to the local client URL.
- Changing instances of "/files/" to http://OpenLabyrinth.mvm.ed.ac.uk/files/ in any OpenLabyrinth resource URLs (such as images).
- Changing certain links so that they open in a new window.
- Changing instances of any HTML escaped characters back to their base format, for instance instances of "'" should be turned back to an apostrophe.

12.6 OpenLabyrinth Client Enhancements

The previous few sections have outlined the underlying basics of creating a remote OpenLabyrinth client service. It is possible however to build additional services on top of this as meet the requirements of the system in which the client service is instantiated. These services may be global to all surfaced OpenLabyrinth cases, they may be attached to specific activities (identified by their unique map ID) or to specific nodes (identified by their unique node ID).

Examples from existing clients include:

- The ability for users to make their own notes either at a particular node or throughout an activity
- The ability for users to record particular kinds/formats of data, for instance when working with a virtual patient it is useful to be able to record several differential diagnoses, to rank them and then to progressively whittle them down to a definite diagnosis as the activity unfolds
- Cross-integration and mashups with other dynamic web applications

13. Customization: Skins and Mashups

13.1 Skins

The skins control how the labyrinth is presented to the user. The skin for a particular labyrinth can be changed in the [Details](#) editor.

Note that, from v3.11 onwards, a completely different format has been used for Skins. They are not compatible with each other. All new skins are created using the new format.

We are currently in transition between the old and new Skin formats, which are not compatible with each other. When you are editing a case and click on 'Skin' on the left side editing menu, you will see the following screen:

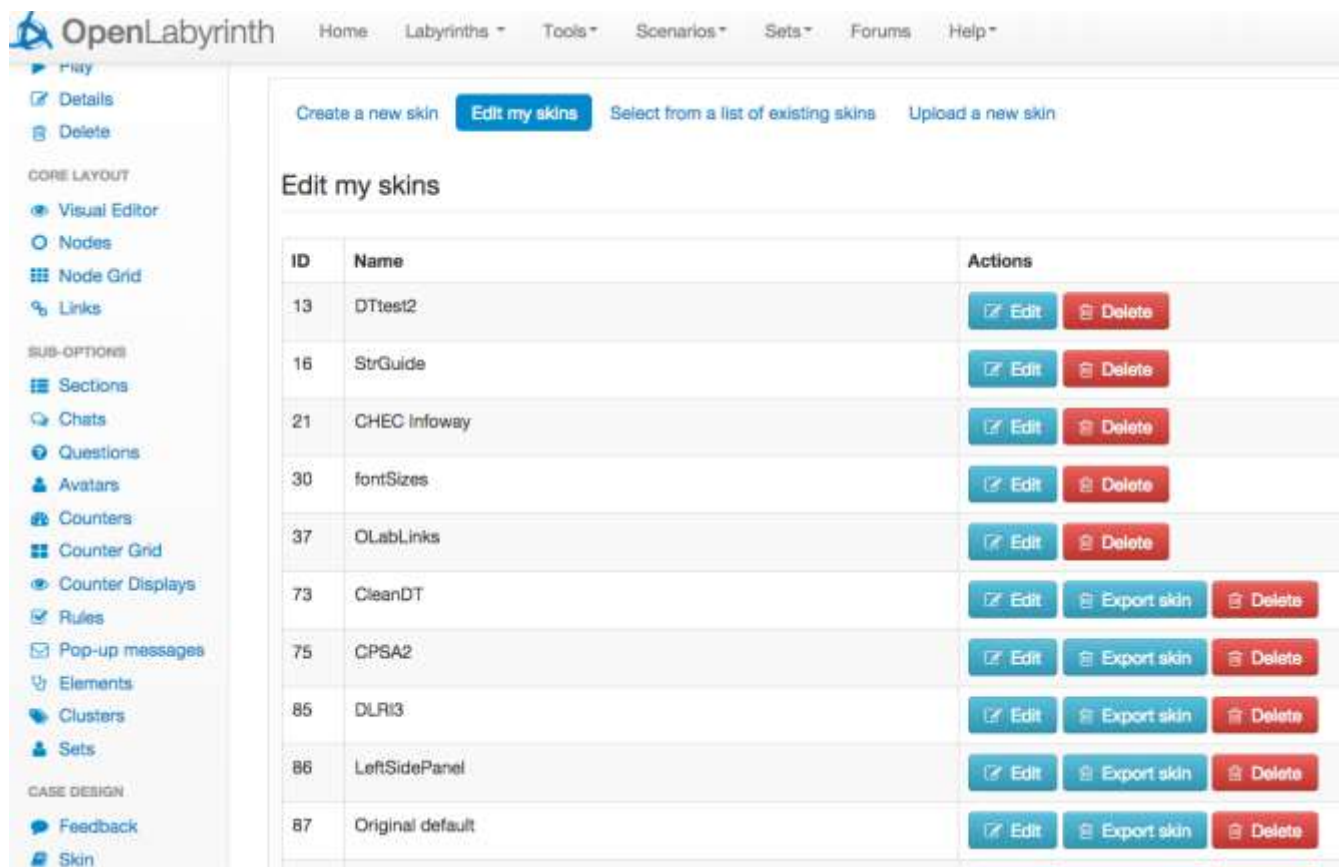


Figure 45: Skin Editor opening screen

The top 5 skins on this page are in the old format and can still be edited to a limited degree. We will first touch on how to work with the old format, for legacy cases. You may be relieved to note that you can still create and edit old format skins with the new Skin Editor. But we encourage all authors to use the new format skins from here on. We will not be supporting old format skins in future versions.

13.2 Working with old format skins

In OpenLabyrinth prior to v3.11, this is what you would see on a typical node, much of which is controlled by the Skin.

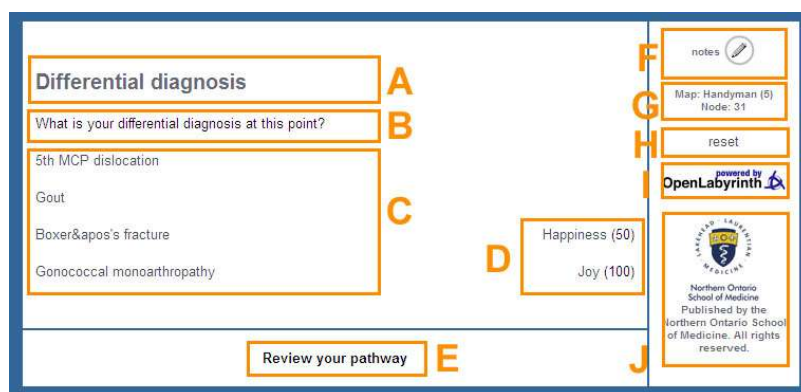


Figure 46: an example OpenLabyrinth skin (for Northern Ontario School of Medicine)

There are a number of different presentation elements in an OpenLabyrinth skin (see figure 46):

A: title	B: message	C: options
D: counters	E: review session	F: notes
G: metadata	H: reset session	I: link to home page
J: skin-specific content		

Effectively all that different skins do is to change the layout and visual presentation of these elements – see figure 47



Figure 47: some example OpenLabyrinth skins

You can edit an existing old format skin, to a limited extent, using the new Skin Editor. You cannot create an old format Skin. When you choose to edit an old format skin, you will see an editor panel like this:

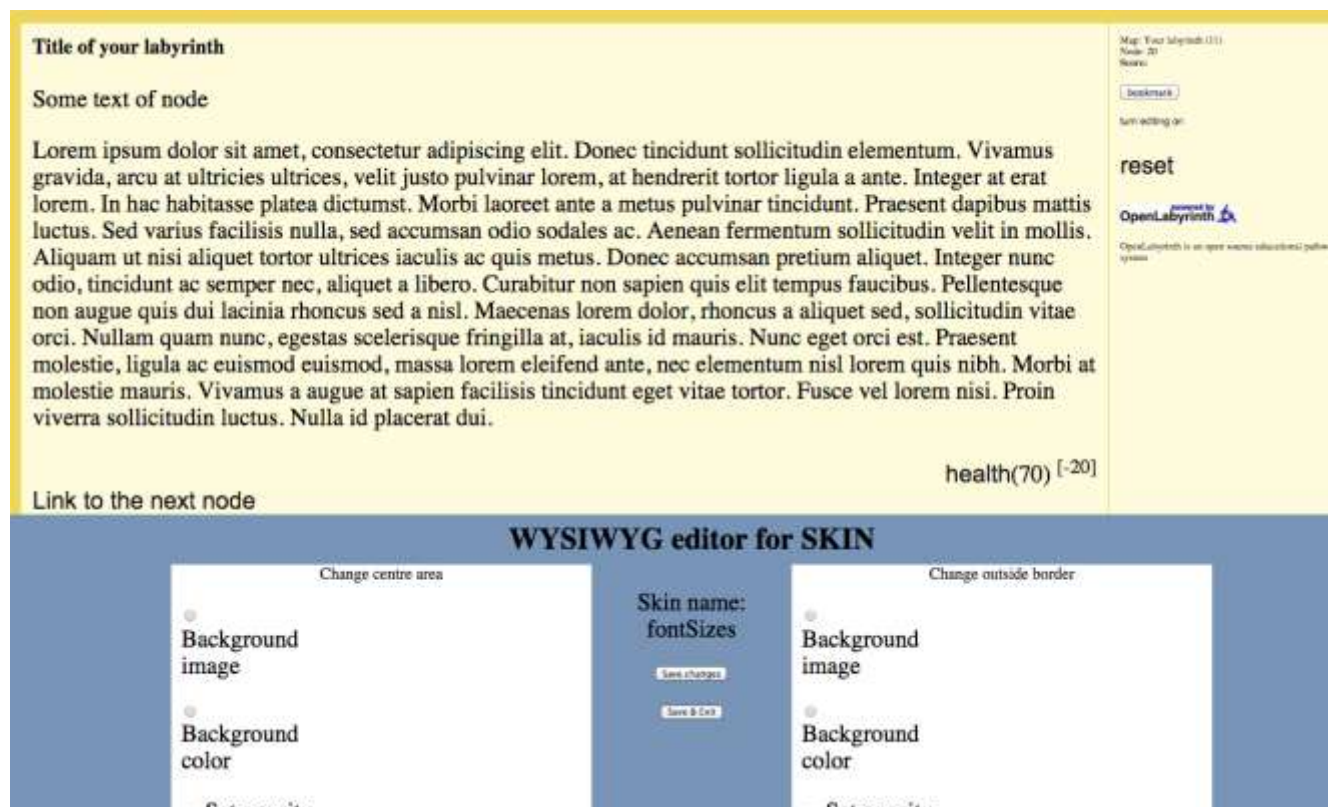


Figure 48: Skin Editor working with an old format Skin

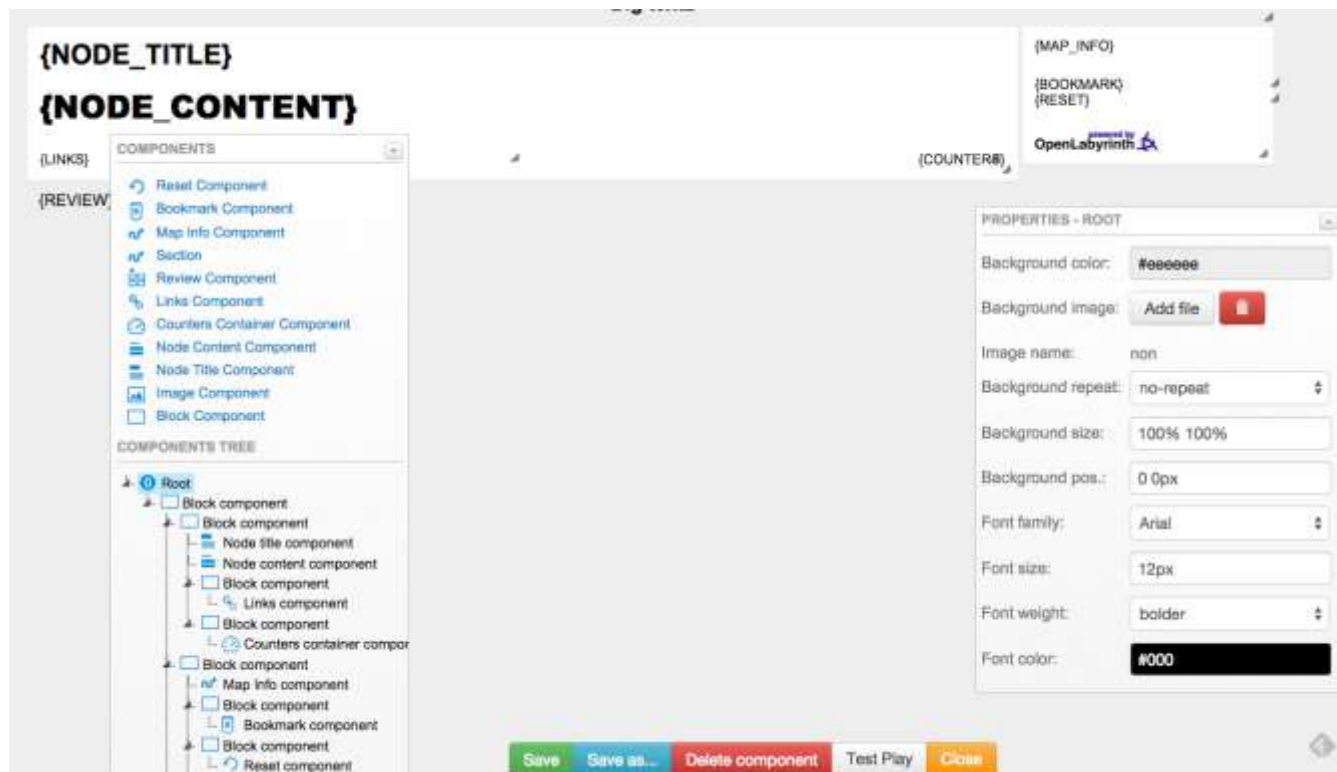
There are only a few things that you can change easily: font sizes, background colors etc. If you are familiar with the vagaries of CSS formatting, you can also manually change how some links behave. Good luck.

13.3 Working with new format Skins

You will see the same opening screen for the Skin Editor as above. New format Skins can be Exported so this is one way to tell if a Skin is in old or new format, without opening the Skin Editor further.

As before, you can create a new skin (which will always be based on the default Basic (new format) Skin), or you can edit one of your existing Skins. Note that you only have access to Skins that you yourself created or imported.

We apologize that the Skin Editor interface, while powerful, is not the most user-friendly. Our rationale was that this is an infrequent task, mostly used by advanced authors, whom we hope will be more tolerant of the quirks of this tool.



In the above screen, you will note the following major items:

- A series of self-explanatory buttons along the bottom
- Two floating panels, components listed in a tree on the left and
- Properties for the selected component on the right
- The main screen shows the current components in a WYSIWYG layout.

You can select the component to work on by clicking on the main screen or by selecting it from the component tree panel. The Properties panel will change to show data for the currently selected component.

One of the easiest things to try first, when getting used to the Skin Editor, is to create a new basic skin and then Delete components that you don't want e.g. Review Component, Bookmark Component, to create a cleaner simpler look.

After that we encourage you to play with the editor to explore how it works. Authors familiar with other programming environments will have little trouble. You can edit most properties and as you change them, the appearance of the master panel will alter to reflect your changes in a fairly accurate WYSIWYG fashion. The [Test Play] button may be useful for more extensive changes.

We have created an example case called 'Skins Game', <http://demo.openlabyrinth.ca/labyrinthManager/global/475> with examples of what the same case can look like with various skins applied. We have embedded the Skin files used with that case, within the case, so that you should be able to Import the case on to another compatible OpenLabyrinth server and try things out for yourself.

13.4 Code

OpenLabyrinth has been written in an open and largely modular way to allow for new developments and changes to the code base. OpenLabyrinth is provided as [open source under a GNU-GPL v3 licence](#) – see Appendix 2, which means that users are free to reuse and adapt the code to meet their own needs. You are strongly encouraged to share any new code with the OpenLabyrinth community. Examples of code changes could include:

- New services such as tracking a differential diagnosis
- New data elements such as those for a different professional discipline
- Integration with other systems such as VLEs, assessment systems or

13.5 Documentation

A copy of this user guide is also provided in PDF and Word DOC format. This can be amended and reused as set out in the Creative Commons Attribution-NonCommercial-ShareAlike 3.0 license (see <http://creativecommons.org/licenses/by-nc-sa/3.0/>). For instance you could add new instructions, translate or adapt it for different contexts.

We have also successfully adapted TiddlyWiki (<http://tiddlywiki.com>), a Javascript-enhanced, single-page wiki, as a context-sensitive help file within some of our case series. Feel free to contact us if you would like further information on how to do this.

13.6 Mashups

OpenLabyrinth remote web services can be used to create mashups with other remote applications. For instance:

- a labyrinth could be used to link with a wiki by pairing its nodes with the wiki pages
- a labyrinth could be used to link with other mashup applications such as Flickr, YouTube or various Google services

See section 11 for more information on using the OpenLabyrinth remote services. See section 23 for more information on connecting with other data systems.

14. Development Techniques

This section pulls together some advice and tips on creating and using OpenLabyrinth activities.

14.1 Using VUE

The use of VUE has been largely superseded for most users by the much improved Visual Editor tool. However, some authors may still like the rapid, flexible and powerful design capabilities of VUE.

VUE is a free topic-mapping tool from Tufts University - see <http://vue.tufts.edu/> for both Windows and Mac. You can use it to create designs for labyrinths by creating boxes to represent nodes and the links between them. Although VUE supports many other features only the boxes (converted to nodes) text in the boxes/nodes and the links (between nodes) will be imported but everything else will be ignored.

VUE allows a whole design to be viewed at once, and to be built in whatever way the author(s) wish (see Figure 49 for some examples). For instance a VUE design could be data projected for a group of authors to comment and build up a map collaboratively. Note that OpenLabyrinth v3 does not support direct import of VUE maps at present. You need to use an intermediate step of importing to an OpenLabyrinth2 server, exporting as an MVP package, which you then Import into OpenLabyrinth. If there is sufficient demand, we will create a VUE import tool for OpenLabyrinth v3 as well.

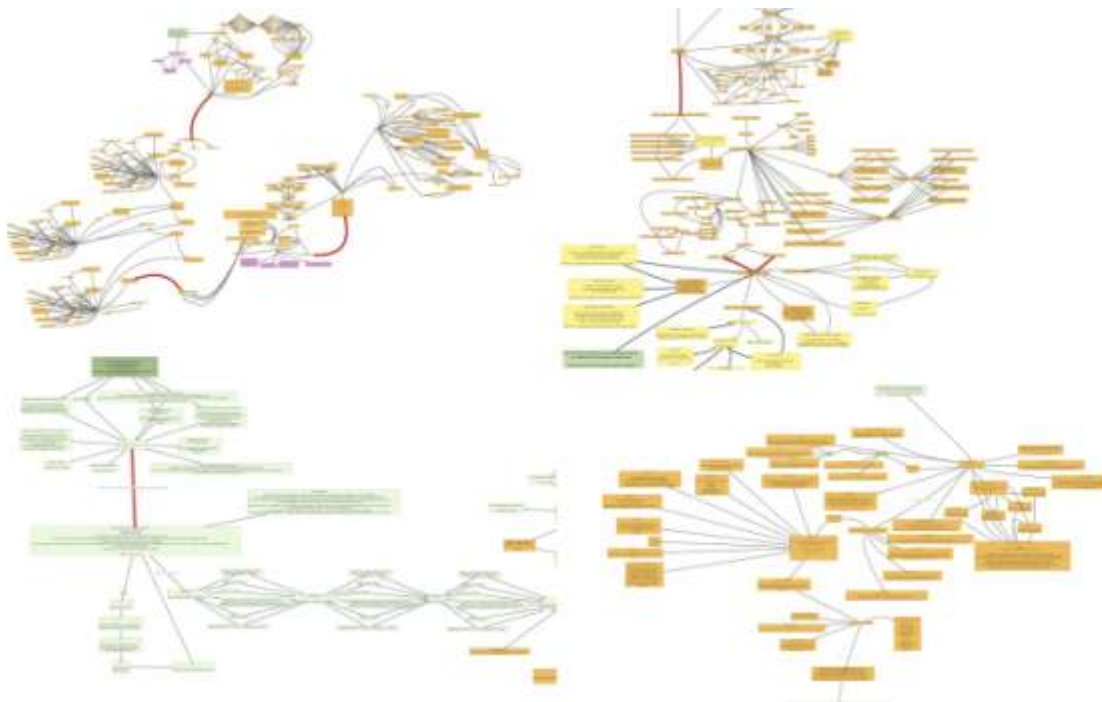


Figure 49: some labyrinth designs created in VUE

As well as creating labyrinth designs in a stepwise manner parts of other designs can be copy-pasted into new designs. For instance a design for an interview or making a diagnosis can be reused across a number of designs. Reusing these 'design patterns' can make development faster and more effective.

14.2 Using Nodes

A node consists of a container for HTML content plus a number of rules and functions. A node may need to be duplicated if one or the other differs. Furthermore, a node may contain a media object that presents an activity in its own right, for instance an assessment question, an animation or an interactive object such as a puzzle or a mini game.

14.3 Using Links

The links represent the topology of the labyrinth activity. You can use these links creatively to assemble different kinds of labyrinths:

- A randomly selected link can be used to represent a dice throw – make six links each for one of the dice values and then set the node to select just the one randomly. When the labyrinth is played the dice throws will be randomly selected each time, use alternative link text to hide what the value is.
- You can use alternative text and icons to control how options are presented to the user. For instance, you can allow users to select the image that best suits the question such as interpreting an x-ray or a histology image

You can use different link types to create more specific designs. For instance, Key Feature Problems (KFPs) can be designed using dropdown menus (Fischer et al 2005).

14.4 Using Counters

You can use counters to track different kinds of values during an activity:

- Costs: a maximum financial budget can be set as the starting value and each choice that involves expenditure can have that cost deducted from the budget.
- Time: a time budget can be deducted from or arbitrary time added to based on the choices made. For instance, selecting a blood pressure test may take 3 minutes while a CT scan may take an hour.
- Psychological factors such as morale, general health or satisfaction can be tracked based on choices made.
- Clinical specific factors such as blood pressure, heart rate or respiratory rate can also be tracked against decisions made.
- Artefacts such as tools, drugs, equipment or keys can be acquired (value increased) and used (value decreased) using counters.

Remember that you can have any number of counters in any one labyrinth.

14.5 Different Kinds of Labyrinth Designs

There are lots of different kinds of designs you can build in Labyrinth:

- Standard professional narratives such as history, examination, investigation, diagnosis and treatment.
- Narratives based on single cases or having to deal with multiple cases at once, each intruding or interrupting the other
- Algorithms such as clinical guidelines, diagnostic pathways for technical or user documentation.
- Script Concordance Testing – this is a powerful method of assessing cognitive pathways and decision making. We encourage authors to explore this approach – simply Google on that phrase and you will find many excellent references from the originating group in Montreal.

15. Collaborative Editing as a Team

It is not hard to put together a simple case as a single author. However, for large projects, it is much better to form a team with various strengths to collaborate on various aspects of case writing. As well as dividing the work, it will also make your cases stronger as you can provide some internal peer review and suggestions about a case.

OpenLabyrinth was not originally conceived as a collaborative authoring environment. Indeed it is possible to seriously mess up each others' work under certain circumstances so some caution is advisable.

15.1 Potential team roles

- Clinician author
We find that having clinicians drive the main focus of a case works well in many situations. But remember that OpenLabyrinth cases do not have to be about patients.
- Learning designer or instructional designer
Not strictly necessary but it is good to have an expert keep your clinicians focused on good learning strategies.
- Copyright clearance and media discovery
This is essential if you are going to make your cases openly available. "Fair Use" provisions do not give educators freedom to use anything they find.
- Learners
It is too easy to lose sight of learner needs. Lots of value in including learners in your team.
- Graphic artist
While there are many wonderful images freely available out there, sometimes the concept you want to convey requires its own picture. Our most productive projects have benefited from an artist.
- Copy editor
Not usually required unless you are posting in a repository
- Peer reviewer
A very good idea. Makes your cases stronger.
- Programmer
Not really needed. But sometimes it helps to have someone who is more familiar with HTML coding.
- System administrator
Your web server will need to be hosted somewhere. OpenLabyrinth can be hosted on a large variety of web servers, including cloud and virtual servers. If your institution or group does not have access to its own server, try contacting one of the OpenLabyrinth servers in your region. Some will be happy to let you try some cases on their server. OpenLabyrinth does not need much administration once it has been installed, your SysAdmin will be pleased to hear.

15.2 General cautions about collaborative editing in OpenLabyrinth

All case information in OpenLabyrinth is stored on a SQL database. But the OpenLabyrinth code has not been written with multi-user authoring in mind. There is no record locking in the database. User access control lists are pretty simple and do not generally prevent multiple authors who all have access to a case from overwriting each others' work inadvertently. We are working on new components in OpenLabyrinth that will make it easier for teams to collaborate. In the meantime, bear in mind the following:

- Visual Editor, Node Grid, Counter Grid – these tools open up the whole case at once. If any one of the team is using one of these tools, it is very easy to overwrite work being done by other team members at the same time.
- Node Grid - This makes it all the more important to remember that you have the whole labyrinth open in the Node Grid editor – beware!

- If you notice that some nodes are changing while you are editing a case, be aware that another author is probably in the case. Avoid the 3 tools above. Try to instant message those whom you think might be in the case.
- Other parts of the OpenLabyrinth editing environment can be more safely used by team members at the same time. For example, if one team member is checking image sources for copyright, and another is creating Questions in the Questions editor, it is unlikely that their work will intersect. But again, be aware that OpenLabyrinth does not prevent two authors from overwriting each others' work.
- Changes made by most parts of the OpenLabyrinth editing environment are usually written to the SQL database right away. This makes it easier to see your changes, even if you are in the middle of previewing or playing a case. But this same effect can throw off another player. Be careful.
- Forums are a useful way for collaborating teams to communicate about a case. Soon, we plan to provide ways of directly linking from case objects into their relevant forums and topics.
- Most objects in OpenLabyrinth have Annotation fields attached. For simple cases and single authors, you will remember what the function of each Rule, Question or Element is. But for more complex cases, this form of annotation or commenting can be invaluable when you or others have to go back to figure out why a case was designed a certain way

Node Grid can be particularly useful in editing a case that has been worked on by others. Nodes are generally created in a logical order and so it is easy to find your way in a labyrinth. But if multiple authors have been involved, over multiple sessions, the logical flow may bear no relation to the NodeID number and logical sections may be interspersed with each other. The logical sorting abilities of the Node Grid editor then become quite attractive to use in team editing.

16. Imagemaps and hotspots

One interesting new feature in OpenLabyrinth v3 is the ability to directly enable hotspots in an image. This is not a new web technique at all, but previously it was a bit fiddly to do in OpenLabyrinth. The Node Editor and specifically, the TinyMCE editor, now directly allows you to create an area within an image that has an embedded hyperlink.

Note that this technique is somewhat dependent on screen size and resolution. It does not translate well to small devices such as smartphones so bear this in mind when designing your case. We generally use an image size of 1024x768 pixels (also known as XGA resolution) because this is the standard screen resolution of a data projector or iPad. It still works on many screen sizes that are larger than this.


First you must use **Node Editor** to do this.

Node Content

Title:

Node Content

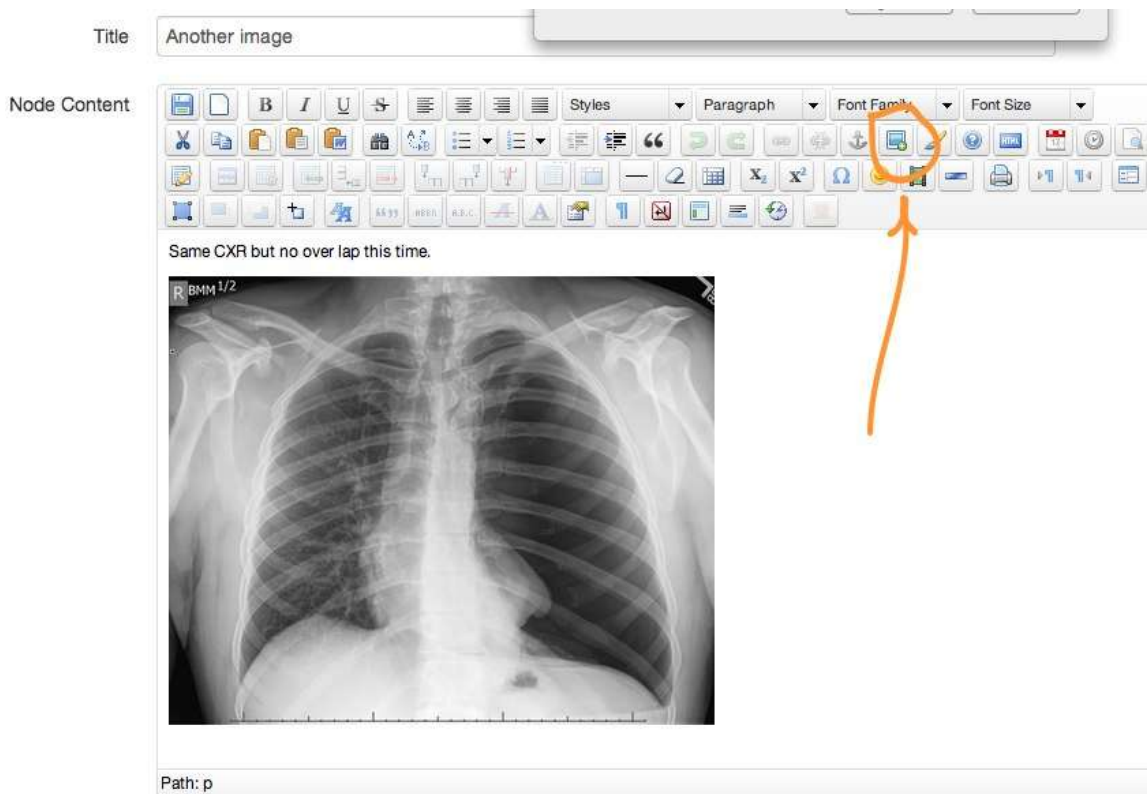
Same CXR but no over lap this time. |



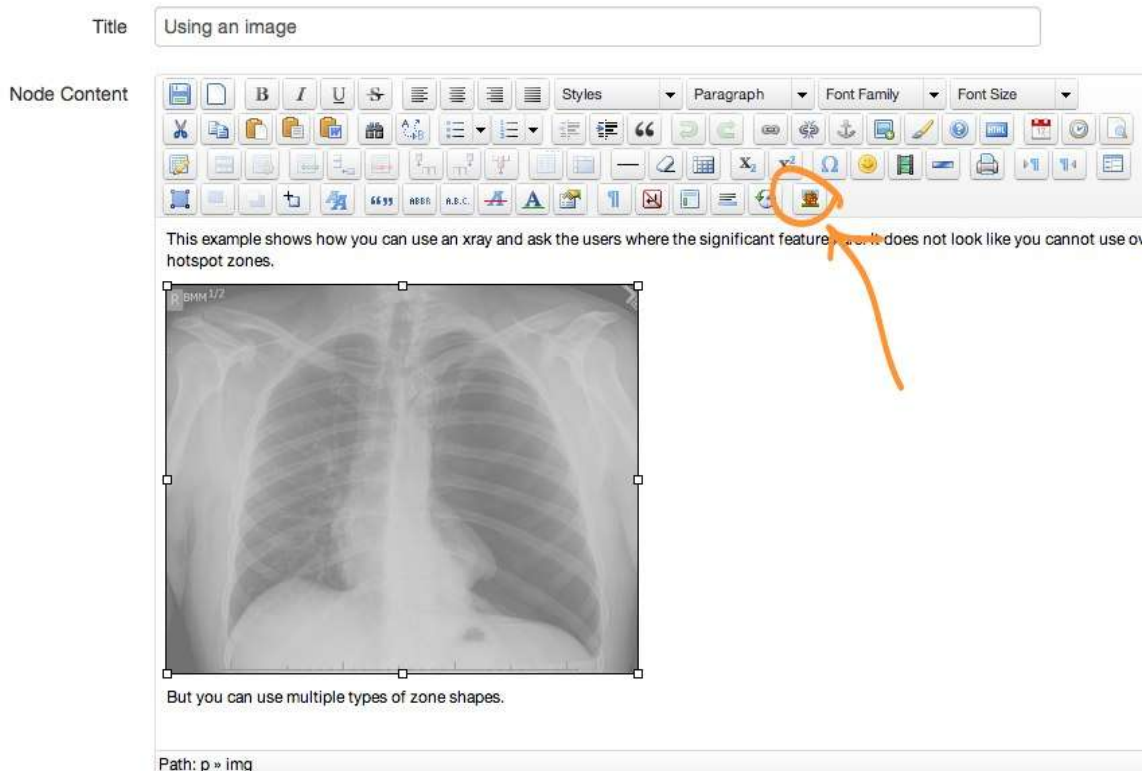
Path: p Words: 8

Supporting Information

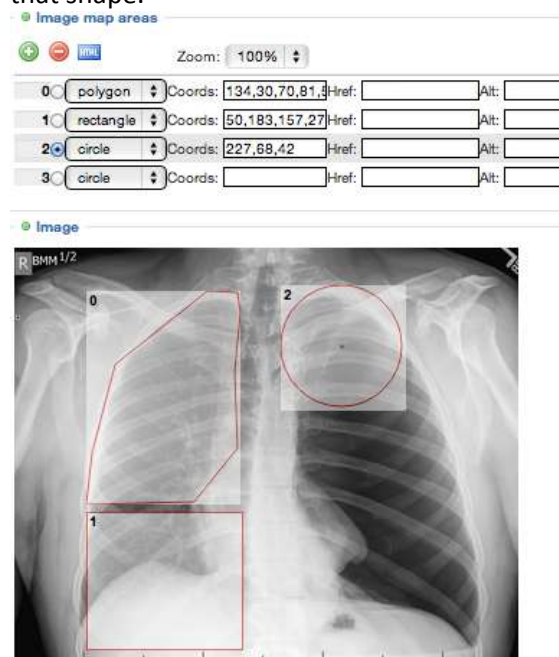
Normally when you are working with an image in OpenLabyrinth, you would insert the wiki style reference, like `[[MR:123]]`. But in this case, we need to tell the TinyMCE editor that is part of the node editor more about the picture you are inserting.



First insert your image, using the image button. Provide a URL to the image. This is easy for images elsewhere on the web. For images in the Files section for the case, you can still do this but you need to insert the local link which will be something like 'files/imageNameHere.jpg' – if you have done it correctly, you will be able to see the picture itself in the TinyMCE editor. Next click on the 'Image Map Editor' button – see below...



This will bring up the editing panel, with your picture in it. You can draw areas of interest, and attach a URL to that shape.



Simply outline the areas that you want to be active hyperlinks and then select the URL that you wish OpenLabyrinth to jump to. This can be another node in the case or outside of OpenLabyrinth entirely. Remember to save when you are done.

With this technique, you can make key areas on your x-ray into teaching points – “spot the fracture” etc.

17. Frequently Asked Questions

In this user guide, we are only highlighting a few common issues. It is much better to maintain a FAQ list on a web site, where they can be more easily updated. Currently, these are hosted at https://sites.google.com/site/openlabyrinth3/support_pages but this is not a very effective system. In future, we will be moving to a more effective system. Other OpenLabyrinth author groups are encouraged to help this open-source project by contributing to this effort.

- 1) Why won't my case run?
The reason it would not play is that there was no starting point or root node set. For snippets, this would not be important but it does make it tough for you to see how it looks. I have set the top node as root and now you can see it Play.
- 2) Why does my case start in the wrong place?
- 3) What are these weird errors we see from Kohana?
- 4) Why do I have to pick from this limited set of choices?
There is no answer there that I like. I want to be able to type in my own answers.
- 5) Why won't OpenLabyrinth run on my browser?
- 6) Where do I find good case examples?
See this section [Virtual Patient case libraries](#)

18. Further Information and Resources

OpenLabyrinth is provided as open source code and as such the authors accept that there may be some bugs and problems but make no guarantees and accept no liabilities for the same. If bugs are found then either fix them and share the code back to the OpenLabyrinth community or make a request for a fix to the community.

Given that Open Labyrinth is community driven, there is no formal user support. OpenLabyrinth community members will generally help each other work through problems but only inasmuch as they have the ability and time to do so. As with any community it takes all members to work together to make the system stronger and better. Suggestions and ideas for new developments should also be circulated to the community.

OpenLabyrinth is licensed under the GNU General Public Licence v. 3.0 – see section 20 Appendix 2: GNU General Public License (GNU-GPL) v. 3.0.

18.1 References

Begg, M., Dewhurst, D. and MacLeod, H. (2005). "Game Informed Learning: Applying computer game processes to Higher Education." *Innovate* 1(6).

Begg, M., Ellaway, R., Dewhurst, D. and Macleod, H. (2007). "Transforming Professional Healthcare Narratives into Structured Game-Informed-Learning Activities." *Innovate* 3(6).

Ellaway, R. (2004). "Modeling Virtual Patients and Virtual Cases." MELD.
http://meld.medbiq.org/primers/virtual_patients_cases_ellaway.htm

Ellaway, R. (2007). *Discipline Based Designs for Learning: The Example of Professional and Vocational Education. Design for Learning: rethinking pedagogy for the digital age.* Beetham, H. and Sharpe, R., Routledge: pp153-165.

Fischer M., Kopp V, Holzer M, Ruderich F, Jünger J (2005) "A modified electronic key feature examination for undergraduate medical students: validation threats and opportunities" [Medical Teacher](#), Volume 27, Issue 5, 2005, Pages 450 – 4

18.2 Virtual Patient case libraries

We all benefit from sharing case materials openly and widely. There are a number of open repositories and libraries available. This is by no means a complete list. Indeed, because libraries are growing and expanding all the time, it is impossible to keep such a list up to date. Various efforts are under way to make cases more discoverable, including semantic linking using the mEducator2 SPARQL discovery engine. In the meantime, we encourage all virtual patient case authors to follow these steps to make your cases more available and more widely used:

- Make your case content open and modifiable using a Creative Commons Attribution-NonCommercial-ShareAlike license. <http://creativecommons.org/licenses/by-nc-sa/3.0/>
- Use an authoring tool that is compliant with the ANSI/Medbiq Virtual Patient standard, which makes it playable on a wider set of players.
- Make your repository searchable by semantic discovery tools such as SPARQL

- Post your cases not just to your own web site, but also to well known repositories such as MedEdPortal.

Case libraries that specifically contain OpenLabyrinth MVP format cases:

- <http://vp.openlabyrinth.ca/> - our own main server
- <http://www.virtualpatients.eu/> - eViP
- <http://pine.nosm.ca/> - Pathways in Narrative Education at NOSM
- <http://fmsharc.cfpc.ca/openlabyrinth/> - the SharcFM shared curriculum in Family Medicine, CFPC
- <http://demo.openlabyrinth.ca/> - our own development server, running the latest OpenLabyrinth code

If you know of other open libraries of compatible cases that should be included here, [please contact us](#).

19. Appendix 1: Installation

OpenLabyrinth is a web application written using PHP and Javascript. You need access to a server that is running a standard LAMP configuration (Linux, Apache, MySQL, PHP), the most common web platform out there, and all completely free and open source. OpenLabyrinth also requires a database joining the code. The most suitable database is MySQL Server. Note that the setup requires a certain amount of technical experience in modifying the server properties of a Linux platform. The whole service can be set up on virtual machine. Mac users can install it on the MAMP virtual machine and run a server on their own laptops.

The latest code and installation instructions can be freely accessed at <https://github.com/OpenLabyrinth/Open-Labyrinth>

19.1 Preparation

Provide a Linux server on which you will first need to install (if not already installed):

You may also need to install:

19.2 Code and Directory

- Download the OpenLabyrinth package from <https://github.com/OpenLabyrinth/Open-Labyrinth> and unzip its contents.

You have now put the code part of OpenLabyrinth in place. The next step is to set up the database.

19.3 Database Setup

You installed MySQL or ensured you had a MySQL database available as part of the preparation step. Note that there are many factors that may change the nature of the following tasks – you should consult a database administrator for help if you get stuck.

Once the database is available:

- Add a new database called 'openlabyrinth'.

Once you have completed these steps that should be OpenLabyrinth all ready to go!

20. Appendix 2: GNU General Public License (GNU-GPL) v. 3.0

Version 3, 29 June 2007

Copyright © 2007 Free Software Foundation, Inc. <<http://fsf.org/>>

20.1 Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

20.2 TERMS AND CONDITIONS

0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically

designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to “keep intact all notices”.
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source.

Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of

violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's “contributor version”.

A contributor's “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to

sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

21. Advanced Programming in OpenLabyrinth

Although the syntax and keywords are quite limited, you can still control OpenLabyrinth with considerable power. At present, you also need patience to work through the parser's quirks but feel free to bounce questions off the user groups – you will find them quite helpful.

21.1 Syntax for counter Rules

In OpenLabyrinth v2, authors had limited ability to regulate or direct the flow of the case narrative or logic. In OpenLabyrinth v3, this has been beefed up considerably by the provision of some simple conditional logic and action words. This is a work in progress and, at present, is quirky and frustrating to work with. But it does open up considerable power if you can be bothered to slog through these quirks.

This section has been left in the Appendices because we anticipate that it will grow considerably, as we provide better examples of how best to use this powerful feature.

Part of the problem of working with these Counter Rules is that currently OpenLabyrinth has an odd double pass through the counter rule. There are various reasons for this. We hope to streamline this aspect of OpenLabyrinth programming in the near future. This second pass through the Rules will often have unanticipated side effects.

21.2 Event triggers and sequence of Counter evaluations

There are a number of events that can alter a Counter value when you are playing a case. It is helpful to understand the sequence in which these events are evaluated by the OpenLabyrinth code, if you are having trouble with getting your Counter Rules to work.

User clicks on linked option → Question in current node alters Counter value → User arrives at node → Node has static value assigned by Counter Grid → Counter Rule is evaluated as true → Counter Rule affects a Counter value → Counter Rule redirects user to another node

See [Double loop problem](#) for a detailed diagram of the algorithm loops that OpenLabyrinth executes when evaluating actions, variables, etc. As you can see, it gets quite difficult to follow until you get used to how it works.

21.3 Basic Syntax

IF.. THEN .. ELSEIF.. ELSE.. ENDIF

GOTO [[NODE:xxx]]

[[CR:nn]] =

Comparator: =, !=, <, <=, >, >=

MATCH([[CR:nn]], "Test string")

UPPER(), LOWER(), PROPER(),

NO-ENTRY

Parentheses work but not consistently.

The Counter Rules editor has two modes of operation. Usually, it is easiest to use the 'Text of rule' tab where you simply use node titles and counter names. But if you are having trouble seeing why a rule is not being parsed correctly, the 'Code of rule' tab can be very useful.

We will provide more examples of Counter Rules and how to use them shortly.

21.4 Parsing text input

Right off the bat, we want to caution you to limit your expectations in how much intelligence you can program into your cases. Accurately interpreting what the User has written is fraught with frustrations. Start simple and build from there. Many authors expect to be able to introduce something close to natural language processing (NLP), with a case that can interpret full sentences.

While the computing world has made great strides in character and voice recognition, we still need computing power in the order of IBM's Watson to generally interpret natural language input in a sensible manner. Well beyond the scope of this software.

Even keeping your expectations low can still lead to problems. For example, say you have three options titled 'chest pain', 'chest wall pain' and 'rib pain'. The possible overlaps of letters makes it quite difficult to construct a set of acceptable strings that will discriminate between possible answers.

```
IF MATCH([[QU_ANSWER]], 'chronic migraine') THEN [[CR:159]] = 10, BREAK;  
IF MATCH([[QU_ANSWER]], 'classic migraine') THEN [[CR:159]] = 7, BREAK;  
IF MATCH([[QU_ANSWER]], 'migraine') THEN [[CR:159]] = 5;
```

The parser will only accept what it is told and cannot make allowances for misspellings, variants or plurals e.g. hemorrhoid, haemorrhoid, hemorrhoids. Piles and piles of variation, one might say. You can however adapt to upper and lower case with the appropriate keywords, most of the time.

In QUESion Rules, you can provide a tiny bit of feedback on the correctness of a response with the CORRECT and INCORRECT keywords eg.

```
IF MATCH([[QU_ANSWER]], 'chronic migraine') THEN [[CR:456]] = 10, CORRECT, BREAK;  
IF MATCH([[QU_ANSWER]], 'classic migraine') THEN [[CR:456]] = 7, BREAK;  
IF MATCH([[QU_ANSWER]], 'migraine') THEN [[CR:456]] = 5, BREAK;  
IF MATCH([[QU_ANSWER]], 'jump up') THEN GOTO [[NODE:10188]], [[CR:456]] = [[CR:456]] + 3,  
BREAK;  
IF MATCH([[QU_ANSWER]], 'jump') THEN GOTO [[NODE:10188]], BREAK;  
IF NOT-MATCH([[QU_ANSWER]], 'migraine') THEN [[CR:456]] = 2, INCORRECT;
```

One of the biggest problems is dealing with blank input, compared to no input. While this sounds trivial, it does cause problems. We are working on ways to fix this – stay tuned.

21.5 Text Validators

You can use the Validator field on free-text input to limit what the user can enter into the field.

For a full list, consult <https://github.com/chriso/validator.js/blob/master/README.md>

- **equals(str, comparison)** - check if the string matches the comparison.
- **contains(str, seed)** - check if the string contains the seed.
- **matches(str, pattern [, modifiers])** - check if string matches the pattern. Either `matches('foo', /foo/i)` Or `matches('foo', 'foo', 'i')`.
- **isEmail(str)** - check if the string is an email.
- **isURL(str [, options])** - check if the string is an URL. options is an object which defaults to {
`protocols: ['http','https','ftp'], require_tld: true, require_protocol: false,`
`allow_underscores: false }`.
- **isAlpha(str)** - check if the string contains only letters (a-zA-Z).
- **isNumeric(str)** - check if the string contains only numbers.
- **isAlphanumeric(str)** - check if the string contains only letters and numbers.
- **isLowercase(str)** - check if the string is lowercase.
- **isUppercase(str)** - check if the string is uppercase.
- **isInt(str)** - check if the string is an integer.
- **isFloat(str)** - check if the string is a float.
- **isNull(str)** - check if the string is null.
- **isDate(str)** - check if the string is a date.
- **isAfter(str [, date])** - check if the string is a date that's after the specified date (defaults to now).
- **isBefore(str [, date])** - check if the string is a date that's before the specified date.
- **isCreditCard(str)** - check if the string is a credit card.

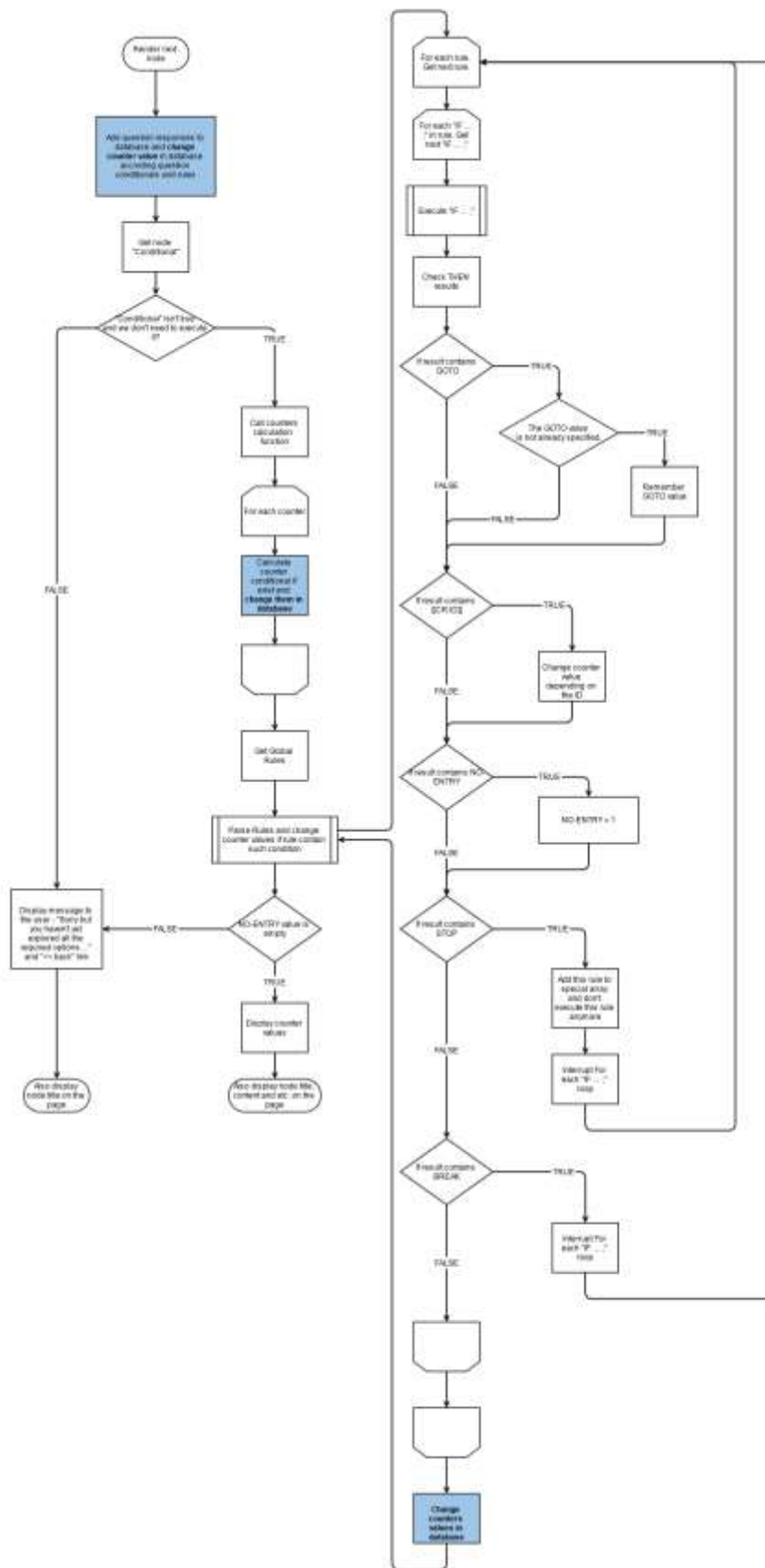
For example, the following Question has an email validator. Note that it only checks from the correct format of an email address. It does not check whether the email identity itself is real and valid.

Stem	<input type="text" value="What is your email address?"/>
Width	<input type="text" value="50"/>
Prompt text <small>Text will automatically appear in response area. Use to give learner a hint or further instruction.</small>	<input type="text" value="press Enter to validate"/>
Rule	<input type="text"/>
Show submit button	<input type="button" value="Show"/> <input checked="" type="button" value="Do not show"/>
Private	<input type="checkbox"/>
Used	<input type="text" value="1"/>
Validator	<input type="text" value="isEmail"/>
Validator error message	<input type="text" value="This is not a valid email address"/>

21.6 Double loop problem

One of the most frustrating things that confuses authors trying to write Rules to control the behaviour of OpenLabyrinth is that there are many factors that affect the value of a variable. And it is not always easy to tell in what order these are happening. This is not the same as a regular programming language where the author has close control over what happens when.

When the User leaves a Node in OpenLabyrinth, the web browser and the OpenLabyrinth server have a complicated series of actions and reactions. For simple Rules, you can mostly ignore these. But for more complex Rules, it sometimes pays to have a better understanding of what happens in which order.



21.7 Quirks and Gotchas

OpenLabyrinth has its share of these. We fix them as we can, when time and money allows. As this is an open-source project, we are always grateful for contributions of time and expertise.

Documenting this rapidly changing area also has its challenges. If you don't find the answer you are looking for here, check the Frequently Asked Questions or the Forums. Or feel free to contact the User groups.

Items to include in programming FAQ:

1. Parentheses are respected
2. You can separate lines with semi-colons
3. IF THEN ELSE structures can be nested... sometimes...
4. Single quotes and double quotes are (usually) interchangeable
5. Currently there is no simple way to assign text input to a Counter
6. RegEx and wildcards are not understood

22. Authentication Systems

OpenLabyrinth has a simple, crude authentication and access control mechanism. For small groups, this is quite sufficient. But for those who are using these cases for multiple classes or who are dealing with many users, it is too cumbersome.

OpenLabyrinth has been designed so that it can be integrated with external authentication systems. For our initial work, we have used OpenLDAP as the example. LDAP is an old system but is very widely used. OpenLDAP is free and open-source. Reconfiguring OpenLabyrinth to work with another LDAP system should be easy.

OpenLabyrinth also supports authentication via OAuth,(<http://oauth.net>) a common protocol that is also used a number of large systems such as Google, Facebook, Twitter, LinkedIn, Tumblr, Github, Flickr. Users who login via OAuth are treated as Learners in the system, with Play level access to Open and Closed cases. They cannot access Private cases, nor can they edit cases. See [Users and Groups](#) for more information on levels of user access.

To make your own installation of OpenLabyrinth accessible by these OAuth authenticators, you will need to ask a Superuser to set up your OpenLabyrinth installation with proper ID numbers and shared 'Secrets' for these systems. This is fairly straightforward, especially if your Superuser has set up OAuth services for other applications in your organization already. Within OpenLabyrinth3, you will find the OAuth setup details under Tools | System Settings and then look under the OAuth tab.

Please consult [our OAuth Manual for a step-by-step guide](#) on how to do this, in the Support section of the main OpenLabyrinth web site at <http://openlabyrinth.ca/>

We have also created an IMS-LTI interface, which provides another way of authenticating users into OpenLabyrinth. See xxyyzz

23. Connecting with Other Systems

Increasingly, our data systems are being built so that they can share information with each other. This can be achieved in a number of ways. For example, OpenLabyrinth makes its data and services available via Web Services. See section [OpenLabyrinth Remote Services](#).

We have limited SCORM compliance in our labyrinths. SCORM was designed for very detailed levels of object interactivity but is correspondingly difficult to implement, and has consequently become outmoded. Other groups are currently exploring the feasibility of enhanced SCORM integration in the OpenLabyrinth player.

A more recent standard that affords data interchange between systems is IMS-LTI (Learning Tools Interoperability). We have made OpenLabyrinth LTI compliant with v1.1 of the spec. Initially, OpenLabyrinth will be an LTI Provider, using 'Basic LTI' but we are also exploring the much greater capability afforded by being an LTI 2.0 Consumer, with a much richer connection to Learning Information Systems.

Along similar lines, we are exploring how best to implement the ADL TinCan xAPI to provide more detailed reporting of the learners' experience when navigating a labyrinth. Because OpenLabyrinth already captures highly detailed metrics on user choices, timing, pathways and responses, it will not be hard to incorporate the xAPI. We are interested in working with other groups who are exploring the data analytics afforded by a Learning Record Store such as the Learning Locker, in conjunction with the Medbiquitous xAPI Interest Group (<http://groups.medbiq.org/medbiq/display/XIG/XAPI+Interest+Group+Home>).

Moving well beyond SCORM objects and basic static metadata, we are also building in the capacity for making our cases' data discoverable by semantic indexing engines such as SPARQL via RDF.

In our HVO project, we were able to connect directly to the HVO Savoir platform, affording direct connectivity and discoverability with a number of other simulation devices. See <http://hvo.org> for more information on this. We were also able to link OpenLabyrinth with the Laerdal™ SimMan 3G API, passing data directly to and from their mannequins.

This is a rapidly evolving area of research. If you are interested in collaborating on such data connectivity projects, [please contact us](#).

24. Security and OpenLabyrinth

Now that OpenLabyrinth has moved to the LAMP (Linux, Apache, MySQL, PHP) platform, it is simpler to create a secure environment. We have been able to satisfactorily use OpenLabyrinth for fully secure, high stakes, summative assessments. But as with any web server, the security depends on how well you have established your basic authentication, encryption and server security at the outset.

It is outside the scope of this user guide to cover the basic aspects of setting up a secure web server. OpenLabyrinth takes a very standard approach and any web server administrator should be able to set things up in a satisfactory manner.

Things to consider:

- SSH, https data stream encryption
- Who has access to the server itself; who has access to the MySQL database
- Using another SQL database eg MSSQL, SQLite
- Firewalls, DMZ

Depending on the intended function of your OpenLabyrinth server, you may choose to make things more or less open. For example, our server at <http://demo.openlabyrinth.ca> has been set up for quite open access by a broad variety of user groups and developers. Accordingly, its security is quite weak and we make no guarantee about uptime, stability of service etc.

On the other hand, our examination servers are locked down behind tight internal firewalls in a DMZ and have no external access. Some of our intermediate servers can be easily accessed by learners but detailed development work is only accessible through a VPN tunnel. We mention this as an example of the many ways in which OpenLabyrinth can be set up, depending on the needs of the target group.

Within the OpenLabyrinth internal structures, there are many different ways to provide access to users, authors and groups. For more information on this, see [Users and Groups](#), [Labyrinth Details](#) and [Scenarios](#).

25. Table of Figures

Figure 1: a typical labyrinth screen	8
Figure 2: a typical labyrinth screen's elements	9
Figure 3: nodes and links.....	13
Figure 4: a node can show more than one InfoButton	15
Figure 5: edit InfoButtons using the Supporting Information field	16
Figure 6: first step in wizard to Create a case	18
Figure 7: select the kind of pattern you want to use	19
Figure 8: Select number of nodes then fill in bare essentials	20
Figure 9: using the Visual Editor to continue mapping the pattern.....	21
Figure 10: adding a small side branch using Visual Editor.....	22
Figure 11: add remaining components here	23
Figure 12: inline node editing:	25
Figure 13: the OpenLabyrinth properties editor, with common actions in sidebar on Left side.	25
Figure 14: the Visual Editor	29
Figure 15: Nodes panel with 'Add a node' button highlighted	31
Figure 16: the WYSIWYG tool bar.....	32
Figure 17: a typical labyrinth screen's elements:	35
Figure 18: node editor page with a linked image.....	36
Figure 19: node editor page showing button to insert an image	37
Figure 20: select the image in Node Editor then click ImageMap button shown.....	38
Figure 21: areas can be circles, rectangles or polygons	39
Figure 22: Node Grid editor works like a table	40
Figure 23: node with 2 Chats before they are clicked	43
Figure 24: same node with 2 Chats after each is clicked.....	44
Figure 25: the Questions editor	45
Figure 26: starting off in creating a Question.....	46
Figure 27: editing a Slider type question	47
Figure 28: inserting a Question into another labyrinth.....	48
Figure 29: editing a single line text entry Question and Rule	51
Figure 30: the avatar editing screen	52
Figure 31: Counter Grid with six Counters	54
Figure 32: screenshot of Node with embedded Counter Display	55
Figure 33: The relationship between VPS, MR, DAM and activity elements in OpenLabyrinth.....	64
Figure 34 an embedded data element in the node editor (left) and how it renders on screen (right)	65

Figure 35: map files editor.	67
Figure 36: the matching editor and the way a matching question is presented to you	70
Figure 37: a typical OpenLabyrinth report histogram.	71
Figure 38: a typical counter trace	72
Figure 39: Scenario Progress view.....	84
Figure 40: the add user screen.....	89
Figure 41: the interface.xml file that can be found in documents/interface.xml	90
Figure 42: the Presentations authoring screens.....	91
Figure 43: OpenLabyrinth Remote Services Architecture.	94
Figure 44: remote components and their interactions with OpenLabyrinth	96
Figure 45: Skin Editor opening screen	100
Figure 46: an example OpenLabyrinth skin (for Northern Ontario School of Medicine)	100
Figure 47: some example OpenLabyrinth skins	101
Figure 48: Skin Editor working with an old format Skin	101
Figure 49: some labyrinth designs created in VUE	104